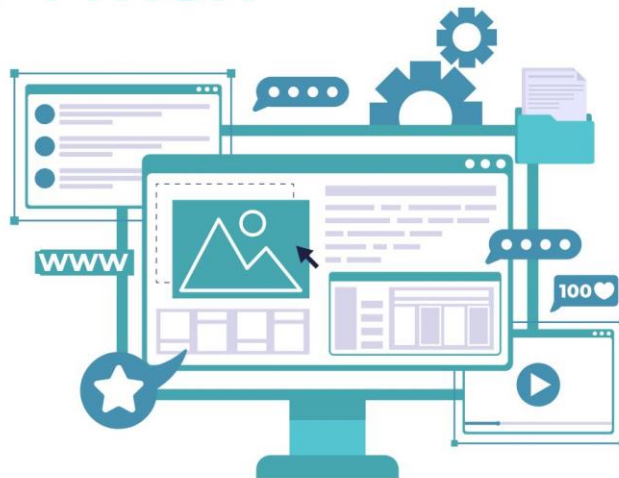


BUKU REFERENSI GRAPHICAL USER INTERFACE (GUI) PADA PYTHON



Christin Erniati Panjaitan
Dini M Hutagalung
Sri Wahyuni Tarigan
Olmes Yosefa Hutajulu
Muhammad Dominique Mendoza

Graphical User Interface (GUI) Pada Python

PENULIS

***CHRISTIN ERNIATI PANJAITAN
DINI M HUTAGALUNG
SRI WAHYUNI TARIGAN
OLNES YOSEFA HUTAJULU
MUHAMMAD DOMINIQUE MENDOZA***

PENERBIT

***UNPRI PRESS
ANGGOTA IKAPI***



**UNIVERSITAS
PRIMA INDONESIA**
UNPRI PRESS

*Hak Cipta dilindungi undang-undang
Dilarang memperbanyak Sebagian atau seluruh isi buku ini dalam bentuk dan
cara apapun tanpa izin tertulis dari penerbit.*

Graphical User Interface (GUI) Pada Python

Penulis : *CHRISTIN ERNIATI PANJAITAN*
DINI M HUTAGALUNG
SRI WAHYUNI TARIGAN
OLNES YOSEFA HUTAJULU
MUHAMMAD DOMINIQUE MENDOZA

Penerbit :
UNPRI PRESS
(ANGGOTA IKAPI)

Alamat Redaksi
Kampus 2
Jl. Sampul No. 4 Medan

BUKU REFERENSI
GRAPHICAL USER INTERFACE (GUI)
PADA PYTHON

Penulis:

Christin Erniati Panjaitan
Dini M Hutagalung
Sri Wahyuni Tarigan
Olnes Yosefa Hutajulu
Muhammad Dominique Mendoza

ISBN:

Penerbit:

UNPRI Press

Redaksi:

Jl. Sampul No.4, Kel. Sei Putih Tengah,
Kec. Medan Petisah, Kota Medan, SUMUT

Cetakan Pertama

Hak Cipta Dilindungi Undang-Undang
Dilarang memperbanyak karya tulis dalam bentuk dan
dengan cara apapun tanpa izin dari penerbit

KATA PENGANTAR

Terlebih dahulu penulis mengucapkan terimakasih kepada Tuhan Yang Maha Esa atas berkatNya sehingga Buku Referensi: Graphical User Interface (GUI) pada Python telah selesai dikerjakan.

Buku referensi ini merupakan panduan yang sangat baik bagi mahasiswa, dosen, peneliti untuk mempelajari konsep dasar Python. Buku ini hanya terdiri dari 2 bab yakni:

- Bab 1 berisikan tentang konsep dasar GUI dan beberapa contoh umum.
- Bab 2 merupakan aplikasi GUI pada pengolahan citra digital.

Ada baiknya memahami dasar pemrograman Python dan Pengolahan Citra sebelum mempelajari buku ini.

Penulis menyadari keterbatasannya dan sangat berterimakasih untuk saran dan kritik agar buku ini dapat terus disempurnakan.

Medan, 5 Februari 2025
Salam Sukses

Penulis

Daftar Isi

KATA PENGANTAR	5
Daftar Isi	6
SINOPSIS	8
Bab 1 Menu Dasar GUI	10
Contoh 1 (Button dan Label).....	10
Contoh 2 (Urutan Button)	11
Contoh 3 (Variasi Button).....	13
Contoh 4 (Username & Password).....	14
Contoh 5 (Konversi USD ke IDR)	14
Contoh 6 (Variasi Label)	16
Contoh 7 (Pendaftaran Peminatan).....	18
Contoh 8 (Relative Positioning).....	20
Contoh 9 (Pilihan Radiobutton)	21
Contoh 10 (CheckBox Penyanyi Favorit).....	22
Contoh 11 (Pesan)	24
Contoh 12 (Scrollbar)	25
Contoh 13 (Listbox).....	28
Contoh 14 (Penskalaan).....	30
Contoh 15 (Progress Bar).....	33
Contoh 16 (Susunan Warna).....	35
Contoh 17 (Check Button).....	42
Bab 2 GUI Pengolahan Citra	45
2.1 Menampilkan Input dan output	46
2.2 Canvas	52
2.3 GUI: RGB to Grayscale	62
2.4 Menampilkan Gambar Tanpa Fungsi	65
2.5 Mengubah Resolusi.....	68

2.6 GUI: RGB to Grayscale to Black and White (BW)	71
Daftar Pustaka	77
Biografi Penulis.....	79
Penulis 1	79
Penulis 2	80
Penulis 3	81
Penulis 4	82
Penulis 5	83

SINOPSIS

Buku ini berisikan konsep Graphical User Interface (GUI) yang menggunakan pemrograman Python. Adapun contoh-contoh yang disertakan pada buku diaplikasikan melalui pemrograman offline. Untuk instalasi IDE Python bisa dipelajari di buku sebelumnya yakni Pengolahan Citra Digital Menggunakan Python. Pada bab 1, buku GUI ini membahas contoh dari pembuatan button dan label. Kemudian contoh selanjutnya, penulis menyampaikan bagaimana membuat dan Menyusun button. Button-button yang dibuat memiliki ukuran yang berbeda satu dan yang lainnya. Lalu pada contoh 3, penulis menampilkan pilihan button secara vertical dan horizontal. Contoh ke-4 merupakan penggunaan harian seperti memasukkan username dan password. Lalu pada contoh ke-5 dan 6 menampilkan proses kalkulasi sederhana seperti konversi mata uang dolar ke rupiah. Pada contoh ke-7 melampirkan contoh proses pendaftaran peminatan mahasiswa Teknik elektro. Kemudian contoh selanjutnya merupakan pembuatan widget yang relative yang dimana ukurannya dapat menyesuaikan dengan tampilan tanpa mengurangi informasi yang ada di dalamnya. Contoh ke-9 menampilkan pilihan dengan gaya radiobutton dan contoh ke-10 menampilkan pilihan dengan checkbox. Suatu pesan dapat juga ditampilkan dalam GUI dan dijelaskan pada contoh ke-11. Contoh ke-12 dan 13 menampilkan pilihan dengan gaya scrollbar dan listbox. Penskalaan dapat ditampilkan dengan vertikal dan horizontal pada contoh ke-

14. Kemudian progress bar menampilkan proses suatu system yang dijelaskan pada contoh ke-15. Contoh ke-16 menampilkan cara menyusun warna, sehingga dapat dikreasikan seperti chessboard. Dan terakhir adalah pilihan dengan menggunakan check button. Pada bab kedua dibahas khusus penggunaan GUI pada pengolahan citra digital. Contoh-contoh yang disertakan pada bab kedua sangat menolong untuk menampilkan input citra, membangun fungsi yang diperlukan dan menampilkan outputnya.

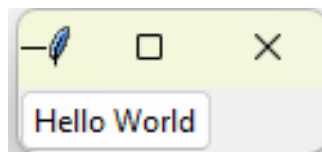
.

Bab 1 Menu Dasar GUI

Graphical User Interface (GUI) merupakan interface yang membantu user mengoperasikan aplikasi yang dipersiapkan. Pada buku ini, akan dibahas GUI menggunakan Python dan ada beberapa library yang dibutuhkan untuk membuat GUI. Tkinter merupakan Tk dan perangkat GUI untuk Tcl/Tk yang dikembangkan oleh John Ousterhout. Pada bab ini akan membahas library **Tkinter**. Tcl merupakan singkatan dari Tool Command Language yang merupakan bahasa skrip yang populer di aplikasi embedded, testing, prototyping dan pengembangan GUI [1]. Program awal 'Hello World' pada Tkinter dapat menggunakan di bawah ini:

```
from tkinter import*  
from tkinter import ttk  
root = Tk()  
ttk.Button(root, text = 'Hello World').grid()  
root.mainloop()
```

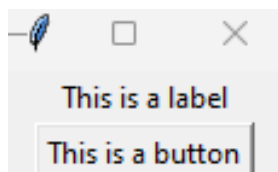
Gambar 1 merupakan tampilan awal pada tkinter.



Gambar 1. Hello World

Contoh 1 (Button dan Label)

Pada contoh 1 akan dibuat program sederhana untuk menampilkan label dan tombol button seperti yang tampak pada gambar 2 di bawah ini.



Gambar 2. Label dan Tombol button

Dimana gambar 2 dapat diperoleh dengan menjalankan kodingan di bawah ini:

#option-1

```
from tkinter import *
root = Tk()
mylabel = Label(root,text="This is a label")
mybutton = Button(root,text="This is a button")
mylabel.pack()
mybutton.pack()
root.mainloop()
```

Atau gambar 2 juga dapat diperoleh dengan menjalankan kodingan di bawah ini:

#Option-2

```
from tkinter import *
root = Tk()
Label(root,text="This is a label").pack()
Button(root,text="This is a button").pack()
root.mainloop()
```

Untuk menggunakan GUI Python maka Tkinter perlu diimport dari library dengan menggunakan perintah: ***from tkinter import **** atau ***import Tkinter***, atau ***import Tkinter as Tk***. Kemudian, memberikan deskripsi label dan button, maka digunakan fungsi **Label** dan **Button** dan kemudian diinput keterangan.

Widget

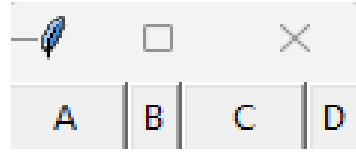
Pada Tkinter terdapat 21 widget yakni: *Toplevel, Canvas, Frame, Menu, OptionMenu, Scale, Text, Label, Checkbutton, Labelframe, Menubutton, PannedWindow, Scrollbar, Bitmap Class, Button, Entry, Listbox, Message, Radiobutton, Spinbox dan Image Class* [2].

Contoh 2 (Urutan Button)

Menampilkan tombol A, B,C dan D secara berurutan dari kiri ke kanan seperti yang tampak pada gambar 3 di bawah ini:



Gambar 3a. Button A, B, C, dan D



Gambar 3b. Button A, B, C, dan D

Adapun gambar 3a dapat diperoleh dari pemrograman di bawah ini:

```
#Memanggil library tkinter
from tkinter import*
#Mempersiapkan root window
root = Tk()
#Mengatur jumlah button
Button(root, text="A").pack(side=LEFT, expand=YES, fill=X)
Button(root, text="B").pack(side=LEFT, expand=YES, fill=X)
Button(root, text="D").pack(side=RIGHT, expand=YES, fill=X)
Button(root, text="C").pack(side=RIGHT, expand=YES, fill=X)
#Mengakhiri program
root.mainloop()
```

Pada gambar 3a, urutan button disesuaikan dengan arah side yang ditentukan. Saat side LEFT maka urutan button harus dari A ke B, tetapi jika side RIGHT yang dipilih maka urutan button harus dari D ke C [3].

Dan gambar 3b dapat diperoleh melalui kodingan di bawah ini:

```
from tkinter import*
root = Tk()
Button(root, text="A").pack(side=LEFT, expand=YES, fill=X)
Button(root, text="B").pack(side=LEFT, expand=0, fill=X)
Button(root, text="D").pack(side=RIGHT, expand=0, fill=X)
Button(root, text="C").pack(side=RIGHT, expand=YES, fill=X)
root.mainloop()
```

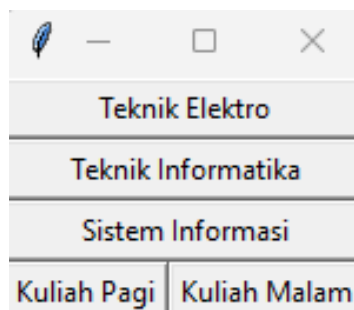
Pada gambar 3b, expand button B dan D dinonaktifkan sehingga button A dan C kelihatan lebih berisi dibandingkan button B dan D. Ada beberapa function yang perlu diketahui jikalau menempatkan lebih dari 1 button pada widget.

- **side** untuk menentukan arah peletakkan dari awal: LEFT, TOP, RIGHT dan BOTTOM.

- **fill** untuk mengisi button sesuai arah: X, Y, BOTH, NONE.
- **expand** untuk mengembangkan button: 1/0 atau YES/NO
- **anchor** untuk menentukan arah peletakkan button: North West (NW), North (N), North East (NE), East (E), South East (SE), South (S), South West (SW), West (W) and CENTER.

Contoh 3 (Variasi Button)

Untuk menampilkan variasi button yang dimana 3 button pertama berurutan dan 1 button terakhir terbagi dua seperti yang tampak pada gambar 4 di bawah ini.



Gambar 4. Variasi button

Adapun gambar 4 dapat diperoleh dengan kodingan di bawah ini:

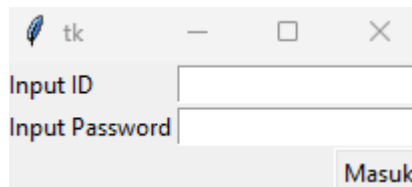
```
from tkinter import*
root = Tk()
Button(root, text="Teknik Elektro").pack(side=TOP, expand=YES, fill=X)
Button(root, text="Teknik Informatika").pack(side=TOP, expand=YES, fill=X)
Button(root, text="Sistem Informasi").pack(side=TOP, expand=YES, fill=X)
Button(root, text="Kuliah Pagi").pack(side=LEFT, expand=0, anchor=W)
Button(root, text="Kuliah Malam").pack(side=LEFT, expand=0, anchor=E)
root.mainloop()
```

Pada gambar 4, side yang dipilih adalah TOP sehingga button yang dituliskan berurutan adalah Teknik Elektro, Teknik Informatika dan Sistem Informasi. Pada button ke 4, button dipecah dua, sehingga side yang dipilih adalah LEFT

dan button pertama yang ditulis adalah Kuliah Pagi kemudian menyusul Kuliah Malam [4].

Contoh 4 (Username & Password)

Latihan selanjutnya adalah membuat tampilan username dan password seperti yang tampak pada gambar 5



Gambar 5. Username dan Password

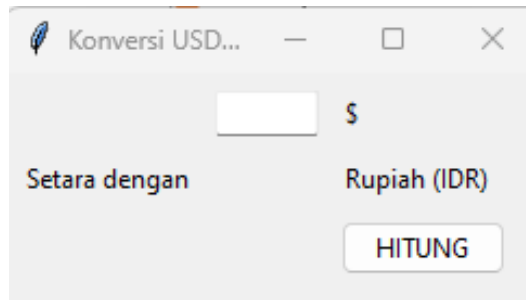
Yang dimana gambar 5 di atas dapat dijalankan dengan menggunakan kodingan di bawah ini:

```
from tkinter import*
root = Tk()
Label(root, text="Input ID").grid(row=0, sticky=W)
Label(root, text="Input Password").grid(row=1, sticky=W)
Entry(root).grid(row=0, column=1, sticky=E)
Entry(root).grid(row=1, column=1, sticky=E)
Button(root, text="Masuk").grid(row=2, column=1, sticky=E)
root.mainloop()
```

Pada kodingan di atas terdapat grid yang dimana GRID merupakan geometry manager yang membagi frame menjadi dua bagian yakni baris dan kolom. Sedangkan fungsi STICKY digunakan untuk memilih lokasi dengan pilihan: N, S, E, W, NW, NE, SW, dan SE.

Contoh 5 (Konversi USD ke IDR)

Pada gambar 6 akan menampilkan frame perhitungan konversi USD ke IDR yang dimana per 11 Januari 2025, 1 USD adalah ±Rp 16.317.



Gambar 6. Konversi USD to IDR

Pada gambar 6 di atas, dapat diperoleh dengan menjalankan kodingan berikut ini:

#Memanggil Library

```
from tkinter import *
from tkinter import ttk
```

#Membuat fungsi perhitungan

```
def calculate(*args):
    try:
        value = float(USD.get())
        IDR.set(value * 16317)
    except ValueError:
        pass
root = Tk()
```

#Judul dan Ukuran Widget

```
root.title("Konversi USD to IDR")
mainframe = ttk.Frame(root, padding="3 3 15 15")
mainframe.grid(column=0, row=0, sticky=(N, W, E, S))
mainframe.columnconfigure(0, weight=1)
mainframe.rowconfigure(0, weight=1)
```

#Membuat variabel untuk input dan output

```
USD = StringVar()
IDR = StringVar()
```

#Membuat dan mengalokasikan Widget

```
feet_entry = ttk.Entry(mainframe, width=7, textvariable=USD)
feet_entry.grid(column=2, row=1, sticky=(W, E))
ttk.Label(mainframe, textvariable=IDR).grid(column=2, row=2, sticky=(W, E))
```

```

ttk.Button(mainframe, text="HITUNG", command=calculate).grid(column=3, row=3,
sticky=W)
ttk.Label(mainframe, text="$").grid(column=3, row=1, sticky=W)
ttk.Label(mainframe, text="Setara dengan").grid(column=1, row=2, sticky=E)
ttk.Label(mainframe, text="Rupiah (IDR)").grid(column=3, row=2, sticky=W)
for child in mainframe.winfo_children(): child.grid_configure(padx=5, pady=5)
feet_entry.focus()

```

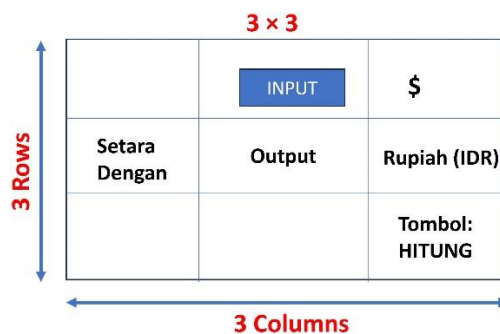
#Menjalankan aplikasi

```

root.bind('<Return>', calculate)
root.mainloop()

```

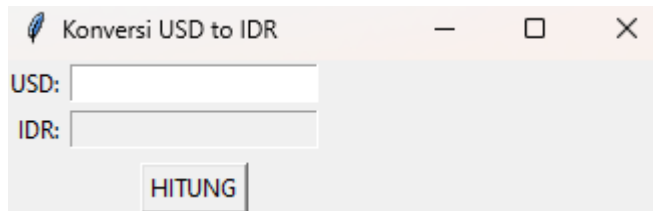
Pada kodingan contoh 5, terdapat suatu fungsi untuk melakukan konversi dari jumlah USD yang diinput. Kemudian untuk penamaan frame menggunakan `root.title('nama')` dan `mainframe` diberikan padding untuk mengatur size saat ditampilkan. USD yang dimasukkan ditampung dalam menu ENTRY dan perhitungan menggunakan BUTTON yang akan diarahkan ke fungsi konversi dengan perintah `calculate`. Tampilan gambar 6 telah diprediksi melalui gambar 7 di bawah ini:



Gambar 7. Layout frame konversi \$ ke IDR

Contoh 6 (Variasi Label)

Dari contoh 5 untuk bentuk yang lebih baik dapat ditampilkan seperti gambar 8 di bawah ini:



Gambar 8. Optional Tampilan dari Contoh 5

Yang dimana gambar 8 dapat diperoleh dari kodingan di bawah ini:

#Memanggil Library

```
from tkinter import*
from tkinter import ttk
```

#Membuat main aplikasi

```
top = Tk()
top.title('Konversi USD to IDR')
```

#Membuat variabel untuk input dan output

```
USD = StringVar()
IDR = StringVar()
```

#Membuat dan mengalokasikan Widget

```
Label(top, text='USD:').grid(row=0, column=0, sticky='e')
Entry(top, textvariable=USD).grid(row=0, column=1, sticky='we')
```

```
Label(top, text='IDR:').grid(row=1, column=0, sticky='e')
Entry(top, textvariable=IDR, state='readonly').grid(row=1, column=1, sticky='s')
```

#Melakukan Perhitungan

```
def calculate(*args):
    try:
        value = float(USD.get())
        IDR.set(f"{value * 16317:,.2f}")
    except ValueError:
        IDR.set('Invalid Input')
```

```
Button(top, text='HITUNG', command=calculate).grid(column=1, row=2, pady=5,
sticky='s')
```

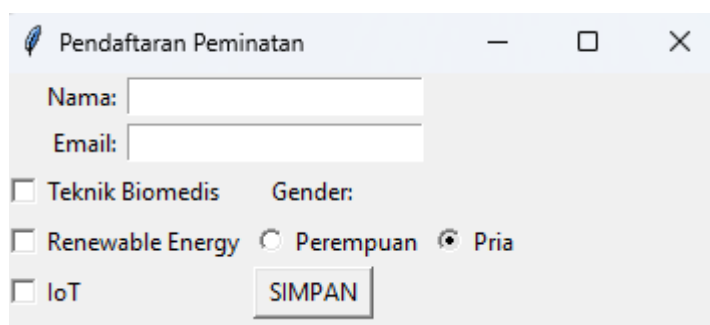
#Mengikat fungsi perhitungan

```
top.bind('<Return>', calculate)
```

#Menjalankan aplikasi
top.mainloop()

Sebelum membangun GUI, harus dibayangkan terlebih dahulu GUI yang akan dibangun butuh berapa baris dan kolom karena akan menggunakan GRID. Pada contoh 6 maka bentuk GUI nya adalah 3 x 2. Pada contoh 6, library yang dibutuhkan dipanggil seperti tkinter lalu, memberi penamaan kepada aplikasi GUI yang akan dibangun. Kedua variable diinput ke dalam tipe data StringVar yakni USD dan IDR. Label USD diposisi baris dan kolom pertama dan tempat pengisiannya (ENTRY) berada di baris pertama kolom ke dua. Label IDR diposisi baris ke dua dan kolom pertama, dan tempat pengisiannya (ENTRY) berada di baris dan kolom ke dua. Ada satu fungsi yang dibangun untuk melakukan perhitungan sederhana konversi dari USD ke IDR yakni variable VALUE dikalikan dengan rate USD saat ini (Rp 16.317). Tombol HITUNG diposisi baris ke tiga dan kolom ke dua. Ada fungsi PADY dan PADX yang merupakan bawaan dari GRID untuk menyediakan padding disekitar widget.

Contoh 7 (Pendaftaran Peminatan)



Gambar 9. Pendaftaran Peminatan

Contoh 7 merupakan tampilan untuk pendaftaran peminatan mahasiswa elektro terhadap 3 bidang yakni Teknik Biomedis, Renewable Energy dan

Internet of Things (IoT). Adapun struktur GUI pada gambar 9 terdiri dari LABEL dan ENTRY. Kemudian ada 2 bentuk button yakni CHECKBUTTON dan RADIOBUTTON. Gambar 9 dapat diperoleh dengan menjalankan kodingan di bawah ini:

#Memanggil Library

```
from tkinter import*  
top = Tk()
```

#Membuat Judul Widget

```
top.title('Pendaftaran Peminatan')  
Label(top,text="Nama:").grid(row=0, column=0, sticky='e')  
Entry(top).grid(row=0,column=1,padx=2,pady=2,sticky='we',columnspan=10)  
Label(top, text="Email:").grid(row=1, column=0, sticky='e')  
Entry(top).grid(row=1,column=1,padx=2,pady=2,sticky='we',columnspan=10)
```

#Pilihan Peminatan

```
Checkbutton(top, text='Teknik Biomedis').grid(row =2, column=0, columnspan=4,  
sticky='w')  
Checkbutton(top, text='Renewable Energy').grid(row =3, column=0, columnspan=4,  
sticky='w')  
Checkbutton(top, text='IoT').grid(row =4, column=0, columnspan=4, sticky='w')
```

#Pilihan Gender

```
Label(top, text="Gender:").grid(row=2, column=4, sticky='we')  
Radiobutton(top, text='Perempuan', value=1).grid(row=3, column=4,  
columnspan=6, sticky='we')  
Radiobutton(top, text='Pria', value=1).grid(row=3, column=15, columnspan=2,  
sticky='we')
```

#Tombol Simpan

```
Button(top, text="SIMPAN").grid(row=4, column=4, sticky='ew', padx=2)
```

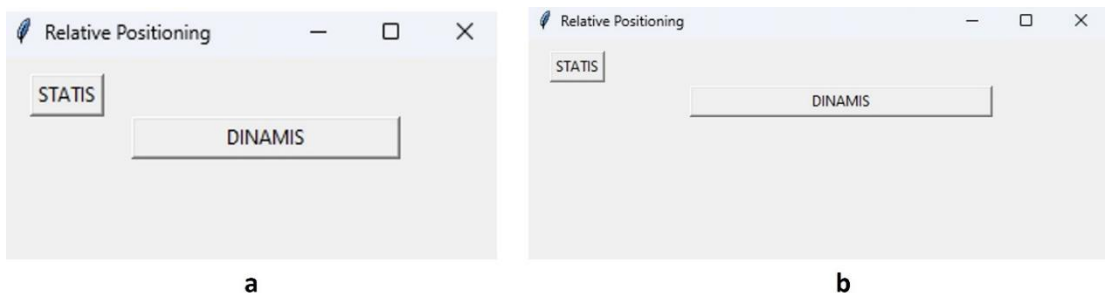
#Akhir Program

```
top.mainloop()
```

Tampilan gambar 9 berbentuk 5 x 16 yang 2 label pertama yakni NAMA dan EMAIL di baris pertama dan kedua pada kolom 1. Terdapat columspan pada masing-masing entry untuk lebar isian data. Kemudian pilih peminatan menggunakan CHECKBUTTON yang dilokasikan pada baris 3, 4 dan 5 pada

kolom pertama dengan masing-masing columnspannya 4. Kemudian ada menu gender dengan RADIOBUTTON. Penamaan gender dilokasi baris ke-3 dan kolom ke-5 sedangkan pilihan perempuan dan pria sejajar. Pilihan perempuan berada pada baris ke-4 dan kolom ke-5 sedangkan pilihan pria pada baris ke-4 kolom ke-16. BUTTON untuk simpan dilokasikan di bawah gender pada baris ke-4 dan kolom ke-4.

Contoh 8 (Relative Positioning)



Gambar 10. Relative Positioning

Gambar 10 merupakan contoh dari tampilan relative positioning yang dimana tombol STATIS pada gambar 10 a dan b dalam kondisi yang fiks walaupun dilakukan pembesaran widget. Sedangkan tombol DINAMIS pada gambar 10 a dan b dapat berubah tergantung ukuran widget. Gambar 10 dapat diperoleh dengan kodingan di bawah ini.

#Memanggil Library

```
from tkinter import*  
root = Tk()
```

#Penamaan Widget

```
root.title('Relative Positioning')
```

#Button posisi statis

```
Button(root, text='STATIS').place(x=20, y=10)
```

#Button posisi dinamis

```
Button(root, text='DINAMIS').place (relx=0.8, rely=0.2, relwidth=0.5, width=10,  
anchor = NE)
```

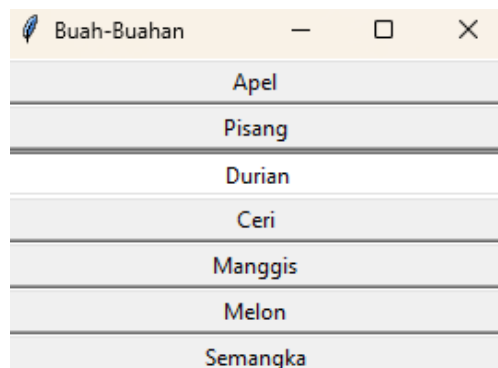
#Akhir Program

```
root.mainloop()
```

Untuk merealisasikan relative positioning maka dibutuhkan relx, rely dan relwidth sedangkan pada button statis fungsi tersebut tidak ada yang ada hanyalah x dan y saja.

Contoh 9 (Pilihan Radiobutton)

Pada gambar 11 di bawah ini menampilkan pemilihan buah-buahan.



Gambar 11. Pilihan Buah-buahan

Adapun gambar 11 dapat diperoleh melalui kodingan di bawah ini:

#Memanggil Library

```
from tkinter import*
```

```
root = Tk()
```

#Memberikan nama Widget

```
root.title('Buah-Buahan')
```

#Memberikan isian pilihan

```
var = IntVar()
```

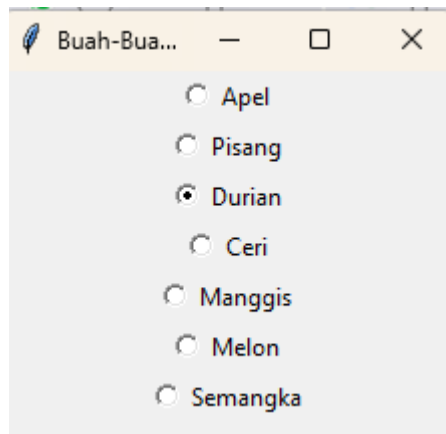
```
for text, value in [('Apel', 1), ('Pisang', 2), ('Durian', 3), ('Ceri', 4), ('Manggis', 5), ('Melon',  
6), ('Semangka', 7)]:
```

```
Radiobutton(root, text=text, value=value, variable=var, indicatoron=0).  
pack(anchor = W, fill=X, ipadx=18)  
var.set(3)
```

#Mengakhiri Pemrograman

```
root.mainloop()
```

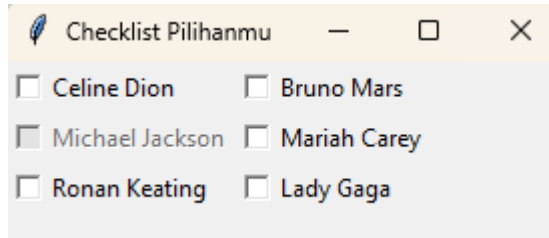
Pada gambar 11 di bawah ini, pilihan buah-buahan menggunakan IntVar() yang disimpan dalam variable var. Lalu nama buaha-buahan diurut dari 1 hingga 7 dalam bentuk text. Pada gambar 11, indicatoron diset 0 agar tampilannya membuat user mengklik baris sedangkan jikalau indicatoron diset 1 maka tampilannya seperti menu radiobutton. Gambar 12 merupakan indicatoron yang diset 1 dan secara tampilan jadi kurang teratur.



Gambar 12. Indicatoron = 1

Contoh 10 (CheckBox Penyanyi Favorit)

Pada gambar 13 di bawah ini merupakan tampilan dari pilihan 6 penyanyi:



Gambar 13. Pilihan Penyanyi

Gambar 13 di atas dapat diperoleh dari kodingan di bawah ini:

```
#Memanggil Library
from tkinter import *
```

```
#Inisiasi Window Utama
```

```
root = Tk()
root.title('Checklist Pilihanmu')
```

```
#Membuat var.Dictionary untuk menyimpan data
var_dict = {}
```

```
# Mendefinisikan pilihan
```

```
cast_members = [
    ('Celine Dion', 0, 0, NORMAL),
    ('Bruno Mars', 0, 1, NORMAL),
    ('Michael Jackson', 1, 0, DISABLED),
    ('Mariah Carey', 1, 1, NORMAL),
    ('Ronan Keating', 2, 0, NORMAL),
    ('Lady Gaga', 2, 1, NORMAL)
]
```

```
#Membuat Checkbox
```

```
for cast_member, row, column, status in cast_members:
    var_dict[cast_member] = IntVar()
    Checkbutton(
        root,
        text=cast_member,
        state=status,
        anchor=W,
        variable=var_dict[cast_member]
    ).grid(row=row, column=column, sticky=W)
```

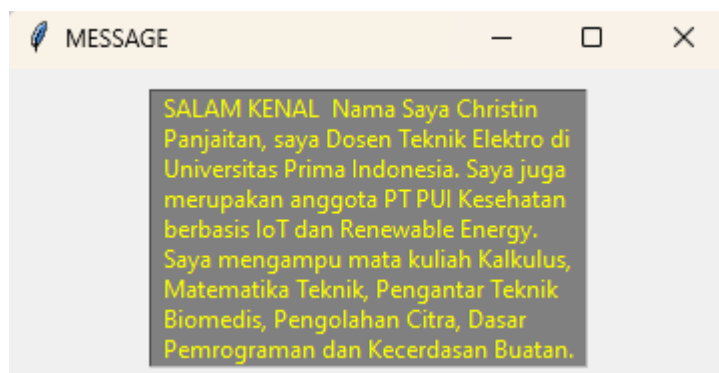
```
#Mengakhiri Program
```

root.mainloop()

Jikalau diperhatikan pada gambar 13, pilihan Michael Jackson tidak aktif sedangkan pilihan yang lain aktif dikarenakan pada kodingan sudah DISABLED. Enam pilihan penyanyi disimpan dalam dictionary dan diurutkan berdasarkan 3 baris x 2 kolom. Gambar 13 menggunakan Checkbutton yang settingannya ada pilihan penyanyi, statusnya baik NORMAL atau DISABLED.

Contoh 11 (Pesan)

Menu yang berisikan pesan saja dapat dikelola seperti yang tampak pada gambar 14 di bawah ini:



Gambar 14. Menu Pesan

Adapun gambar 14 di atas dapat diperoleh melalui pemrograman di bawah ini:

```
#Memanggil Library  
from tkinter import*
```

```
#Inisiasi Window Utama  
root = Tk()  
root.title('MESSAGE')
```

```
#Isi Pesan  
Message(root, text="SALAM KENAL ")
```

"Nama Saya Christin Panjaitan, saya Dosen Teknik Elektro di Universitas Prima Indonesia."

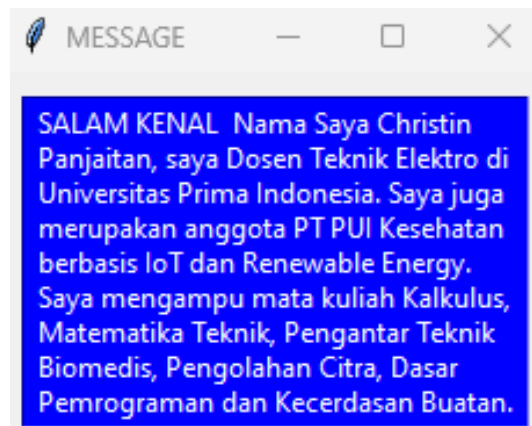
" Saya juga merupakan anggota PT PUI Kesehatan berbasis IoT dan Renewable Energy."

" Saya mengampu mata kuliah Kalkulus, Matematika Teknik, Pengantar Teknik Biomedis, Pengolahan Citra, Dasar Pemrograman dan Kecerdasan Buatan.",
bg='gray', fg='yellow',
relief=GROOVE).pack(padx=10, pady=10)

#Mengakhiri Program

root.mainloop()

Pada gambar 14 di atas, pesan berupa text yang dimana besar windownya adalah 10 x 10. Kemudian ada RELIEF untuk mendefinisikan tampilan text pada border window. Background dan foreground dapat diganti seperti tampak pada gambar 15 di bawah ini. Gambar 15 di bawah ini dengan text yang sama backgroundnya diganti warna biru dan fontnya menjadi warna putih.



Gambar 15 Menu pesan

Contoh 12 (Scrollbar)

Contoh 12 ini membahas tentang scrollbar seperti yang tampak pada gambar 16 di bawah ini:



Gambar 16. Scrollbar

Pada gambar 16 di bawah menampilkan urutan angka dari 0 hingga 19. Tampilan gambar 16 dapat diperoleh melalui kodingan di bawah ini:

#Memanggil Library

```
from tkinter import*
root = Tk()
```

#Memberi nama menu

```
root.title("Scrollbar")
```

#Mengatur ukuran scrollbar

```
list = Listbox(root, height=5, width=15)
scroll = Scrollbar(root, command=list.yview)
list.configure(yscrollcommand=scroll.set)
```

#Mengatur tata letak pilihan dan scroll

```
list.pack(side=LEFT)
scroll.pack(side=RIGHT, fill=Y)
```

#Mengatur angka

```
for item in range(20):
    list.insert(END, item)
```

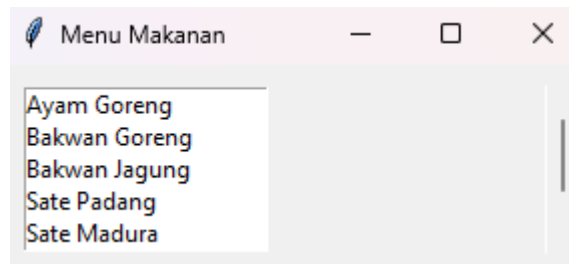
#Mengakhiri Program

```
root.mainloop()
```

Pada kodingan diawali dengan memanggil library tkinter dan memberi nama menu yakni SCROLLBAR. Kemudian variable list ditentukan tinggi dan lebarnya. Kemudian fungsi scrollbar digunakan dan variable list diberdayakan Bersama dalam variable scroll. Pada kodingan list.pack dipilih LEFT dan

scroll.park dipilih RIGHT agar angka tampil sebelah kiri dan scroll sebelah kanan.

Adapun gambar 16 di atas bisa diganti isianannya dengan text seperti yang tampak pada gambar 17 di bawah ini:



Gambar 17. Scrollbar menu makanan

Adapun gambar 17 di atas dapat diperoleh dengan menjalankan program di bawah ini:

```
#Memanggil Library
from tkinter import *

#Membuat root window
root = Tk()

#Memberi nama menu
root.title('Menu Makanan')

#Konfigurasi listbox dan scrollbar
menu_list = Listbox(root, height=5, width=20)
scroll = Scrollbar(root, command=menu_list.yview)
menu_list.configure(yscrollcommand=scroll.set)

#Mengatur layout listbox dan scrollbar
menu_list.pack(side=LEFT, padx=10, pady=10)
scroll.pack(side=RIGHT, fill=Y, padx=5, pady=10)

#Menambahkan isi menu makanan
menu_list.insert(END, 'Ayam Goreng')
```

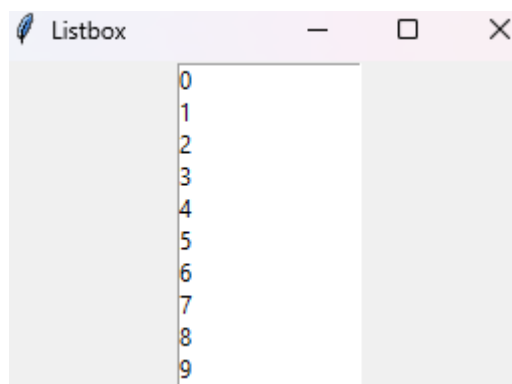
```
menu_list.insert(END, 'Bakwan Goreng')
menu_list.insert(END, 'Bakwan Jagung')
menu_list.insert(END, 'Sate Padang')
menu_list.insert(END, 'Sate Madura')
menu_list.insert(END, 'Bakso Ayam')
menu_list.insert(END, 'Lontong Malam')
```

```
#Mengakhiri Program
root.mainloop()
```

Pada kodingan gambar 17, diawali dengan memanggil library tkinter dan membuat root window. Kemudian dilanjutkan dengan konfigurasi listbox dan scrollbar yang pada gambar 17 tingginya 5 dan lebarnya 20. Variable scroll berisikan command pada list-list menu. Kemudian menu akan ditampilkan pada sebelah kiri dan scroll diatur di sebelah kanan. Lalu menu makanan ditambahkan dengan fungsi insert.

Contoh 13 (Listbox)

Selanjutnya selain scrollbar, listbox juga bisa menampilkan pilihan seperti yang tampak pada gambar 18 berikut ini. Gambar 18 menampilkan urutan angka dari 0 – 9 dan judul windonya adalah LISTBOX.



Gambar 18. Listbox 0 – 9

Gambar 18 tersebut dapat diperoleh dengan menjalankan program di bawah ini:

```

#Memanggil Library
from tkinter import*
root = Tk()

#Memberi nama menu
root.title('Listbox')

#Mengatur list dan ukurannya
list = Listbox(root, width=15)

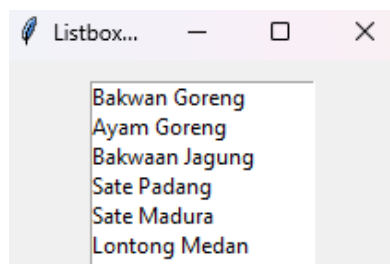
#Mengatur isi list
list.pack()
for item in range(10):
    list.insert(END, item)

#Mengakhiri Program
root.mainloop()

```

Adapun contoh 13 ini pada kodingannya dimulai dengan menginput library tkinter dan memberi nama windownya: listbox. Kemudian fungsinya listbox dipanggil dan diatur ukurannya. Isi dari listbox juga diatur dengan fungsi list.pack dan diisi dengan urutan angka 0-9 yang prosesnya dibantu dengan conditional statement: FOR.

Gambar 18 di atas dapat diganti isinya dengan text seperti tampak pada gambar 19 di bawah ini:



Gambar 19. Listbox string

Gambar 19 dapat diperoleh dengan menjalankan kodingan di bawah ini:

```

#Memanggil library
from tkinter import *

#Membuat root window

```

```

root = Tk()

#Memberikan penamaan window
root.title('Listbox Example')

#Mengatur listbox dan ukurannya
menu_list = Listbox(root, width=20)

#Memasukkan pilihan pada listbox
menu_list.insert(END, 'Bakwan Goreng')
menu_list.insert(END, 'Ayam Goreng')
menu_list.insert(END, 'Bakwaan Jagung')
menu_list.insert(END, 'Sate Padang')
menu_list.insert(END, 'Sate Madura')
menu_list.insert(END, 'Lontong Medan')

#Pack listbox
menu_list.pack(padx=10, pady=10)

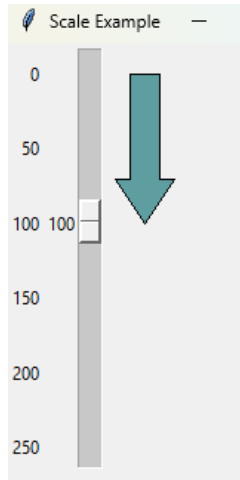
#Mengakhiri Program
root.mainloop()

```

Kodingan pada gambar 19 diawali dengan menghadirkan library tkinter, membuat root window dan penamaan menu. Kemudian ukuran listbox diatur dengan besaran 20. Lalu variable menu_list diikuti dengan fungsi insert menambahkan pilihan-pilihan menu makanan. Selanjutnya menu_list dipaack dengan 10 padx dan 10 pady dan program selesai.

Contoh 14 (Penskalaan)

Keuntungan dari scale adalah user dapat menentukan nilai dari yang terendah ke paling tinggi sehingga bisa linear. Contoh penskalaan dapat dilihat pada gambar 20 di bawah ini:



Gambar 20. Penskalaan

Adapun contoh 14 dapat diperoleh dengan menjalankan kodingan di bawah ini:

```
#Memanggil library
from tkinter import *
```

```
#Mempersiapkan root window
root = Tk()
```

```
#Memberi judul window
root.title("Scale Example")
```

```
#Membuat Fungsi Untuk Mengatur ukuran
```

```
def set_height(canvas, height_str):
    height = int(height_str) # Convert string to integer
    height = height + 21 # Add offset to the height
    y2 = height - 30 # Calculate y2 coordinate
    if y2 < 21: # Ensure y2 does not go below 21
        y2 = 21
    #Koordinate Polygon
    canvas.coords(
        'poly',
        15, 20, 35, 20, 35, y2, 45, y2, 25, height, 5, y2, 15, y2, 15, 20
    )
    #Koordinat Garis
    canvas.coords(
        'line',
        15, 20, 35, 20, 35, y2, 45, y2, 25, height, 5, y2, 15, y2, 15, 20
```

```
)
```

#Membuat Canvas

```
canvas = Canvas(root, width=50, height=300, bd=0, highlightthickness=0)
canvas.create_polygon(0, 0, 1, 1, 2, 2, fill='cadetblue', tags='poly')
canvas.create_line(0, 0, 1, 2, 2, 0, 0, fill='black', tags='line')
```

#Membuat Skala Vertikal

```
scale = Scale(
    root,
    orient=VERTICAL,
    length=284,
    from_=0,
    to=250,
    tickinterval=50,
    command=lambda h: set_height(canvas, h)
)
```

#Meletakkan skala pada Canvas

```
scale.grid(row=0, column=0, sticky='NE')
canvas.grid(row=0, column=1, sticky='NWSE')
```

#Inisial Value

```
scale.set(100)
```

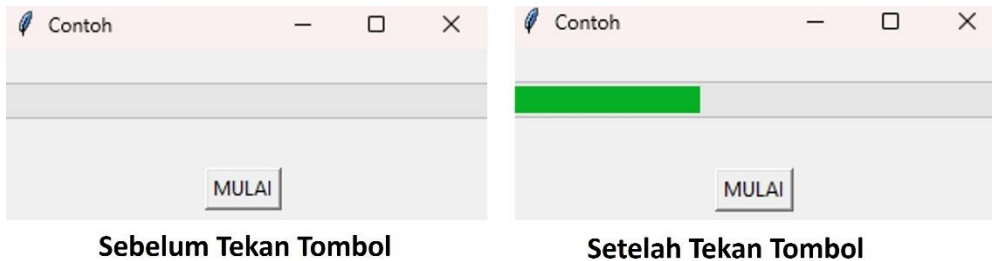
#Mengakhiri Program

```
root.mainloop()
```

Dalam penskalaan di contoh 13, diawali dengan memanggil library, mempersiapkan root pada window dan memberika judul window. Kemudian fungsi dibuat yang menampung 2 variable yakni canvas dan height_str. Pada variable height, maka inputan angka langsung diubah menjadi angka int dan ditambahkan 21. Variable y2 merupakan pengurangan height dengan 30. Saat y2 lebih kecil dari 21 maka y2 akan diseterakan dengan 21. Lalu koordinat polygon dan garis diatur.

Contoh 15 (Progress Bar)

Progress bar merupakan widget yang diperlukan untuk menampilkan proses yang sedang berlangsung seperti yang tampak pada gambar 21 di bawah ini:



Gambar 21. Progressbar

Gambar sebelah kiri merupakan tampilan sebelum tombol ditekan dan gambar sebelah kanan merupakan tampilan setelah tombol ditekan. Gambar 21 dapat diperoleh melalui kodingan berikut ini:

#Memanggil Library

```
import tkinter as tk
from tkinter import ttk
import time
```

#Fungsi Progress

```
def start_progress():
    progress.start()

    for i in range(101):
        time.sleep(0.05)
        progress['value'] = i
        root.update_idletasks()
    progress.stop()
```

```
root = tk.Tk()
root.title("Contoh")
```

#Widget Progressbar

```
progress = ttk.Progressbar(root, orient="horizontal", length=300,
mode="determinate")
progress.pack(pady=20)
```

#Tombol Button

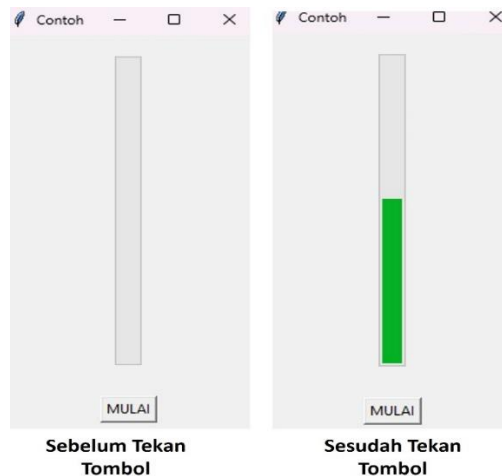
```
start_button = tk.Button(root, text="MULAI", command=start_progress)
start_button.pack(pady=10)
```

#Mengakhiri Program

```
root.mainloop()
```

Gambar 21 merupakan contoh dari progressbar yang dimana dalam kodingan library yang dibutuhkan dipanggil. Kemudian fungsi progress dibangun yakni def start_progress() yang dimana di dalam fungsi tersebut ada fungsi FOR untuk range i di 101, time sleep di 0.05 detik dan progress valuenya adalah i yang sudah dipanggil pada deklarasi FOR. Menu diberi nama contoh dengan fungsi root.title. Lalu step berikutnya, bentuk progressbar dibuat horizontal dengan Panjang 300. Kemudian tombol diberi nama MULAI dengan pady 10. Progress bar juga dapat dibuat vertikal dengan mengganti di bagian #widget progressbar, pilihan **orient: vertical**

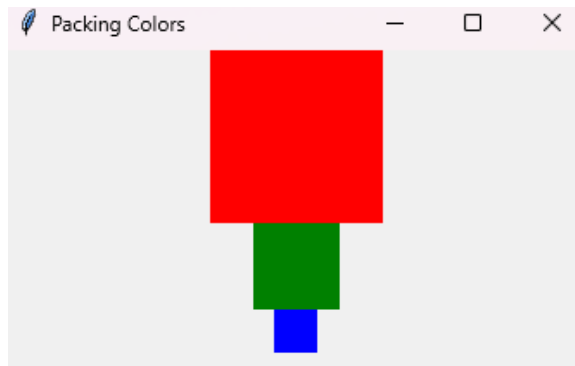
Adapun tampilan progressbar tampak vertical pada gambar 22 di berikut ini. Adapun pada gambar 22 di sebelah kiri merupakan progressbar yang belum ditekan tombol MULAI dan sebelah kanan yang telah ditekan tombol MULAI.



Gambar 22. Progressbar tampak vertikal

Contoh 16 (Susunan Warna)

Contoh selanjutnya dari pemrograman GUI di Python ini adalah menyusun frame dari yang terbesar hingga terkecil dengan variasi warna seperti yang tampak pada gambar 23 di bawah ini [5].



Gambar 23. Menyusun warna

Adapun gambar 23 sebelumnya dapat diperoleh dengan kodingan di bawah ini:

#Mengimport library

```
import tkinter as tk
```

#Mempersiapkan Root Window

```
window = tk.Tk()
```

```
window.title('Packing Colors')
```

#Mengatur Frame 1

```
frame1 = tk.Frame(master=window, width=100, height=100, bg="red")
```

```
frame1.pack()
```

#Mengatur Frame 2

```
frame2 = tk.Frame(master=window, width=50, height=50, bg="green")
```

```
frame2.pack()
```

#Mengatur Frame 3

```
frame3 = tk.Frame(master=window, width=25, height=25, bg="blue")
```

```
frame3.pack()
```

#Menyelesaikan program

```
window.mainloop()
```

Diawali dengan memanggil library yang dibutuhkan kemudian mempersiapkan root dengan variabel window dan memberi nama menu yakni Packing Colors. Kemudian frame 1 dengan lebar dan tinggi masing-masing 100 diberikan warna merah, kemudian frame 2 dengan ukuran 50×50 diberikan warna hijau dan frame 3 dengan ukuran 25×25 diberikan warna biru. Dengan kodingan yang sama maka warna bisa diubah seperti yang tampak pada gambar 24.

#Mengatur Frame 1

```
frame1 = tk.Frame(master=window, width=100, height=100, bg="black")  
frame1.pack()
```

#Mengatur Frame 2

```
frame2 = tk.Frame(master=window, width=50, height=50, bg="white")  
frame2.pack()
```

#Mengatur Frame 3

```
frame3 = tk.Frame(master=window, width=25, height=25, bg="black")  
frame3.pack()
```

#Mengatur Frame 1

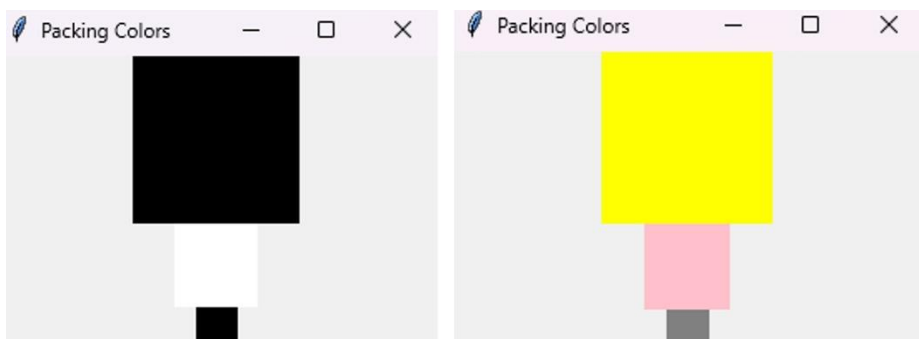
```
frame1 = tk.Frame(master=window, width=100, height=100, bg="yellow")  
frame1.pack()
```

#Mengatur Frame 2

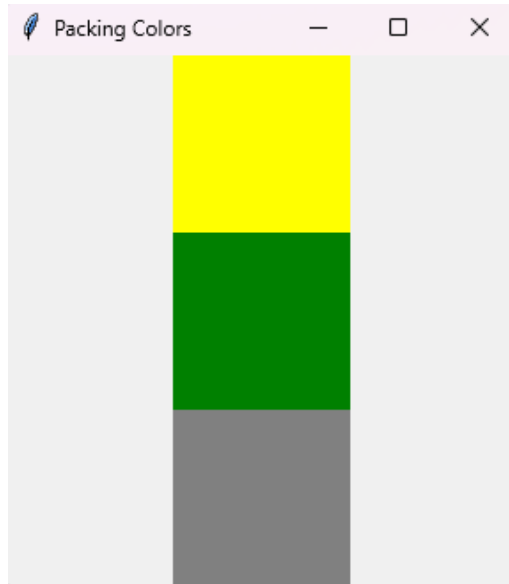
```
frame2 = tk.Frame(master=window, width=50, height=50, bg="pink")  
frame2.pack()
```

#Mengatur Frame 3

```
frame3 = tk.Frame(master=window, width=25, height=25, bg="gray")  
frame3.pack()
```



Gambar 24. Susunan gambar dengna variasi warna



Gambar 25. Susunan gambar dengan ukuran serupa
Gambar 25 merupakan tampilan susunan gambar dengan ukuran yang serupa dari frame 1 ke 3. Adapun modifikasi kodingannya tampak di bawah ini:

#Mengatur Frame 1

```
frame1 = tk.Frame(master=window, width=100, height=100, bg="yellow")
frame1.pack()
```

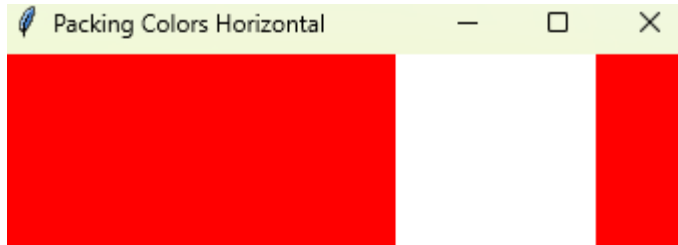
#Mengatur Frame 2

```
frame2 = tk.Frame(master=window, width=100, height=100, bg="green")
frame2.pack()
```

#Mengatur Frame 3

```
frame3 = tk.Frame(master=window, width=100, height=100, bg="gray")
frame3.pack()
```

Pada kodingan di atas terlihat ukuran frame 1 ke 3 memiliki ukuran serupa baik width dan height yakni 100×100 . Pada gambar 26 di bawah ini, susunan warna dapat diatur horizontal.



Gambar 26. Susunan warna horizontal

Adapun kodingan untuk gambar 26 adalah sebagai berikut:

#Memanggil Library

```
import tkinter as tk
```

#Mempersiapkan root window

```
window = tk.Tk()
window.title('Packing Colors Horizontal')
```

#Mengatur Frame 1

```
frame1 = tk.Frame(master=window, width=200, height=100, bg="red")
frame1.pack(fill=tk.Y, side=tk.LEFT)
```

#Mengatur Frame 2

```
frame2 = tk.Frame(master=window, width=100, bg="white")
frame2.pack(fill=tk.Y, side=tk.LEFT)
```

#Mengatur Frame 3

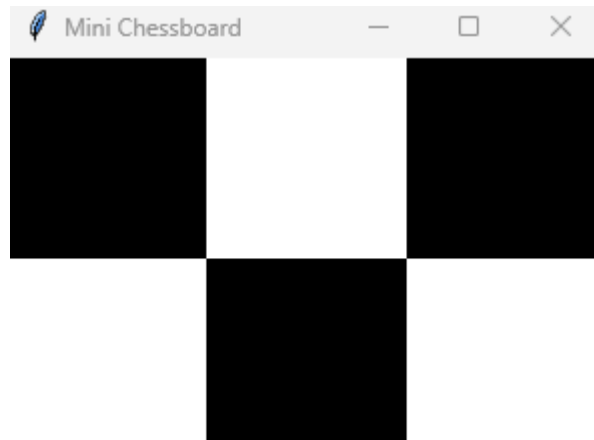
```
frame3 = tk.Frame(master=window, width=50, bg="red")
frame3.pack(fill=tk.Y, side=tk.LEFT)
```

#Menyelesaikan Program

```
window.mainloop()
```

Pada gambar 26, kodingan dimulai dengan memanggil tkinter lalu mempersiapkan root dan penamaan window. Lanjut dengan persiapan frame 1, 2 dan 3. Frame 2 dipersiapkan warna merah dengan lebar 200 dan tinggi 100. Kemudian frame 2 dipersiapkan warna putih dengan lebar 100 dan frame 3 dengan lebar 50 dengan warna merah.

Dengan konsep yang sama juga, dapat dibuat mini chessboard seperti gambar 27 di bawah ini:



Gambar 27. Layout mini chessboard

Gambar 27 ini dapat diperoleh dari pengembangan pemrograman sebelumnya yang hasilnya seperti di bawah ini:

#Memanggil Library

```
import tkinter as tk
from tkinter.ttk import *
```

#Mempersiapkan root window

```
window = tk.Tk()
window.title('Mini Chessboard')
```

#Mengatur Frame 1a

```
frame1a = tk.Frame(master=window, width=100, height=100, bg="black")
frame1a.grid(row=0, column=0)
```

#Mengatur Frame 1b

```
frame1b = tk.Frame(master=window, width=100, height=100, bg="white")
frame1b.grid(row=0, column=1)
```

#Mengatur Frame 1c

```
frame1c = tk.Frame(master=window, width=100, height=100, bg="black")
frame1c.grid(row=0, column=2)
```

#Mengatur Frame 2a

```
frame2a = tk.Frame(master=window, width=100, height=100, bg="white")
frame2a.grid(row=1, column=0)
```

#Mengatur Frame 2b

```
frame2b = tk.Frame(master=window, width=100, height=100, bg="black")
```

```
frame2b.grid(row=1, column=1)
```

#Mengatur Frame 2c

```
frame2c = tk.Frame(master=window, width=100, height=100, bg="white")  
frame2c.grid(row=1, column=2)
```

#Menyelesaikan Program

```
window.mainloop()
```

Adapun gambar 27 dengan kodingan di bawahnya dimulai dengan menghadirkan tkinter, mempersiapkan root window dan membuat nama menu. Kemudian dengan fungsi grid, diatur akan berapa baris dan kolom yang digunakan. Gambar 27 terlihat membutuhkan 3 baris dan 3 kolom untuk membangun mini chessboard. Dimulai dari baris pertama dan kolom pertama diisi warna hitam, baris pertama dan kolom kedua diisi warna putih dan seterusnya.

Untuk membuat kotak catur yang lebih besar dengan ukuran 8×8 maka bisa menggunakan conditional statement seperti pemrograman berikut ini:

Memanggil Library

```
import tkinter as tk
```

Mempersiapkan root window

```
window = tk.Tk()  
window.title('Chess Board')
```

Ukuran kotak catur

```
balok = 50
```

Membuat papan catur dengan nested loop

```
for row in range(8): # 8 baris  
    for col in range(8): # 8 kolom  
        # Tentukan warna berdasarkan posisi baris dan kolom  
        color = "black" if (row + col) % 2 == 0 else "white"  
        # Membuat kotak catur  
        frame = tk.Frame(  
            master=window,  
            width= balok,
```

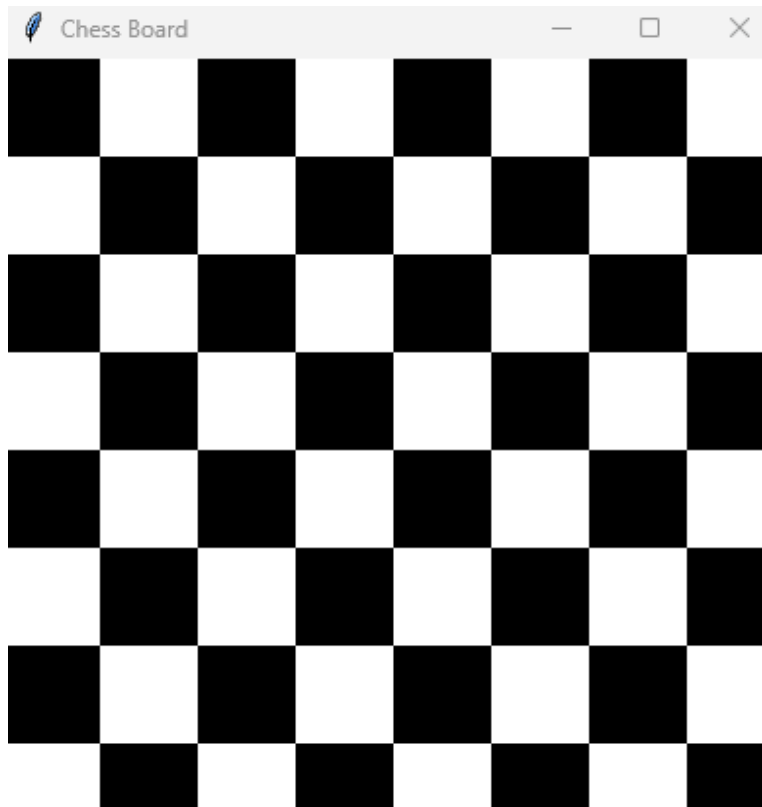
```

        height=balok,
        bg=color
    )
    # Menempatkan kotak pada grid
    frame.grid(row=row, column=col)

# Menyelesaikan Program
window.mainloop()

```

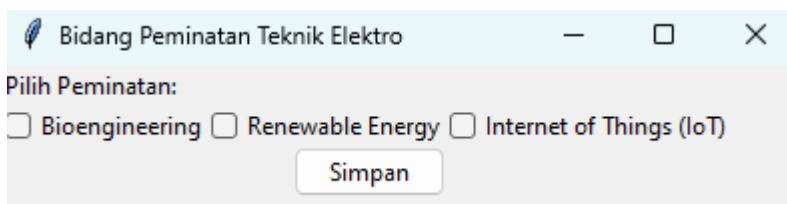
Setelah memanggil library tkinter, mempersiapkan root dan memberikan nama, kemudian ukuran baloknya baik baris dan kolom disetting. Pada gambar 28, ukuran baloknya masing-masing 50×50 . Kemudian fungsi FOR dipanggil untuk baris terlebih dahulu dalam range 8 dan kemudian dilanjutkan dengan kolom dengan range 8 juga. Kemudian warna hitam akan diberikan jikalau penjumlahan row dan col hasilnya dibagi 2 sama dengan 0 selain itu akan diberikan warna putih. Kemudian fungsi grid digunakan untuk mengurutkan balok sesuai susunan yang hasilnya seperti tampak pada gambar 28 berikut ini:



Gambar 28. Chessboard 8×8

Contoh 17 (Check Button)

Pada contoh 17 berikut ini akan menampilkan pemilihan bidang peminatan Teknik elektro dengan format yang berbeda yakni gabungan dari check button, label dan button seperti yang tampak pada gambar 29 di bawah ini:



Gambar 29. Bidang Peminatan Teknik Elektro

Adapun gambar 29 dapat ditampilkan melalui pemrograman di bawah ini:

#Memanggil Library

```
from tkinter import *  
from tkinter.ttk import *
```

#Mempersiapkan Window

```
window = Tk()  
window.title("Bidang Peminatan Teknik Elektro")  
window.geometry('400x75')
```

#Mempersiapkan Label

```
Label(window, text="Pilih Peminatan:").grid(row=0, sticky=W)
```

#Mempersiapkan Checkbox 1

```
chk_state = BooleanVar()  
chk_state.set(False)  
chk = Checkbutton(window, text='Bioengineering', var=chk_state)  
chk.grid(column=0, row=2)
```

#Mempersiapkan Checkbox 2

```
chk_state2 = BooleanVar()  
chk_state2.set(False)  
chk2 = Checkbutton(window, text='Renewable Energy', var=chk_state2)  
chk2.grid(column=1, row=2)
```

#Mempersiapkan Checkbox 3

```
chk_state3 = BooleanVar()  
chk_state3.set(False)  
chk3 = Checkbutton(window, text='Internet of Things (IoT)', var=chk_state3)  
chk3.grid(column=2, row=2)
```

#Mempersiapkan Button

```
Button(window, text="Simpan").grid(row=10, column=1)
```

#Menyelesaikan Program

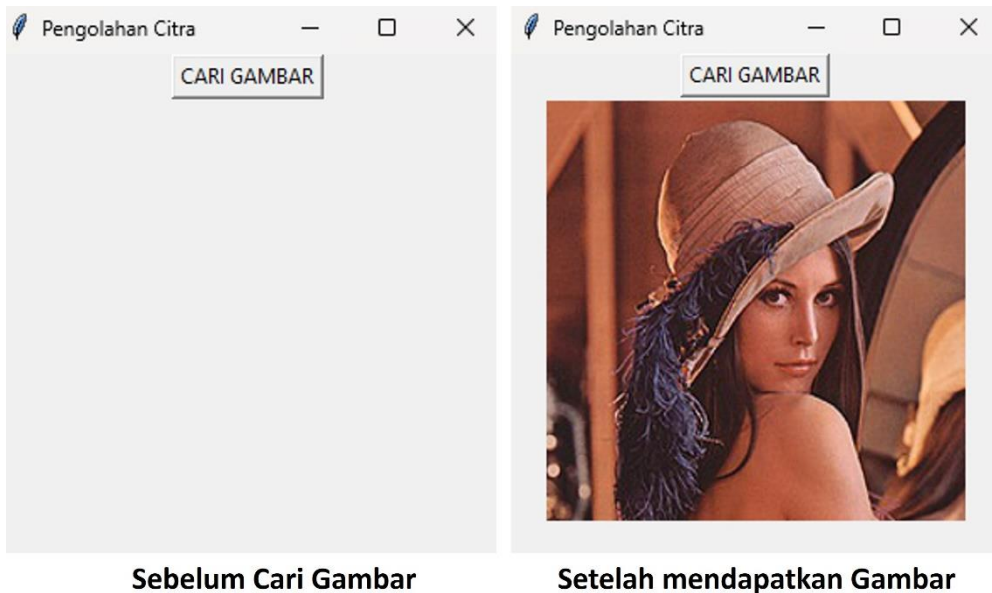
```
window.mainloop()
```

Dimulai dengan memanggil library tkinter dan ttk. Lalu mempersiapkan window baik root, penamaan dan ukurannya. Lalu dilanjutkan dengan label, yang dimana label dibantu dengan fungsi grid untuk mengatur tata letaknya

pada baris pertama. Lalu lanjut dengan checkbox pertama yang dimana variable `check_state` hingga `check_state3` diisi dengan `BooleanVar()` dan diset menjadi `False` sehingga tampilannya kosong dan butuh user untuk checklist. Lalu setiap checkbox diberi nama dari Bioengineering, Renewable Energy, Internet of Things (IoT) dan diatur tata letak pada baris dan kolomnya. Terakhir, tombol button diatur akan diposisikan pada baris ke 10 di kolom ke dua.

Bab 2 GUI Pengolahan Citra

Library Tkinter dipergunakan penulis untuk menampilkan proses pengolahan citra secara professional menggunakan GUI seperti tampak pada gambar 30 di bawah ini:



Gambar 30. Menampilkan Gambar Menggunakan GUI

Adapun tampilan di atas dapat diperoleh dengan menjalankan kodingan di bawah ini:

#Memanggil Library

```
from tkinter import *  
from PIL import ImageTk, Image  
from tkinter import filedialog  
import os
```

#Mengatur ukuran

```
root = Tk()  
root.geometry("300x300")  
root.resizable(width=True, height=True)  
root.title('Pengolahan Citra')
```

#Membangun Fungsi

```
def openfn():
    filename = filedialog.askopenfilename(title='PENGOLAHAN CITRA')
    return filename
def open_img():
    filename = openfn()
    img = Image.open(filename)
    img = img.resize((250, 250))
    img = ImageTk.PhotoImage(img)
    panel = Label(root, image=img)
    panel.image = img
    panel.pack()
```

#Mempersiapkan Tombol

```
btn = Button(root, text='CARI GAMBAR', command=open_img).pack()
```

#Menyelesaikan Program

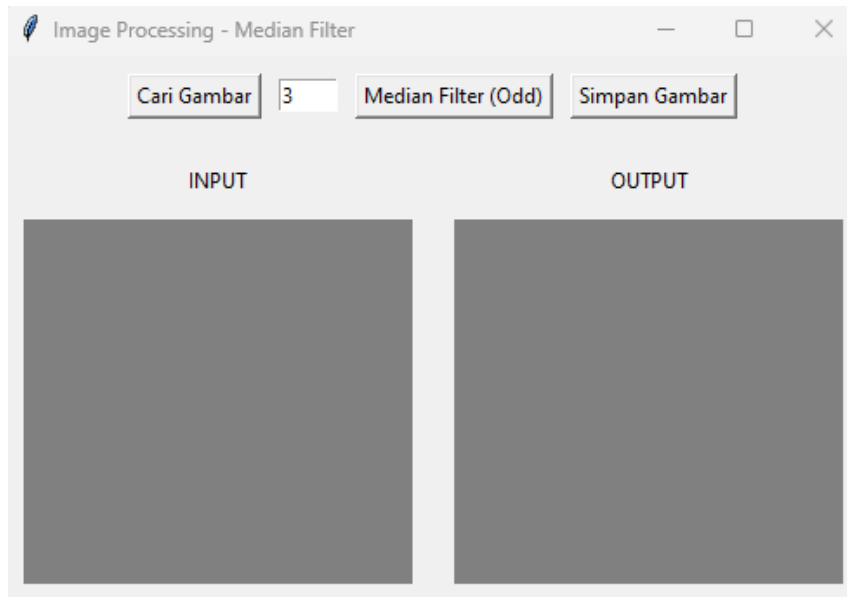
```
root.mainloop()
```

Mengaplikasikan GUI pada pengolahan citra dimulai bagaimana menampilkan citra terlebih dahulu. Diawal program, library yang dibutuhkan dipanggil seperti tkinter, PIL, dan os. Kemudian root window dipersiapkan, besaran windownya ditentukan yakni 300×300 dan memberikan penamaan window: Pengolahan Citra. Lalu sebuah fungsi dibangun untuk mencari gambar di folder dan mengatur resolusinya. Lalu tombol button dibangun dan program selesai.

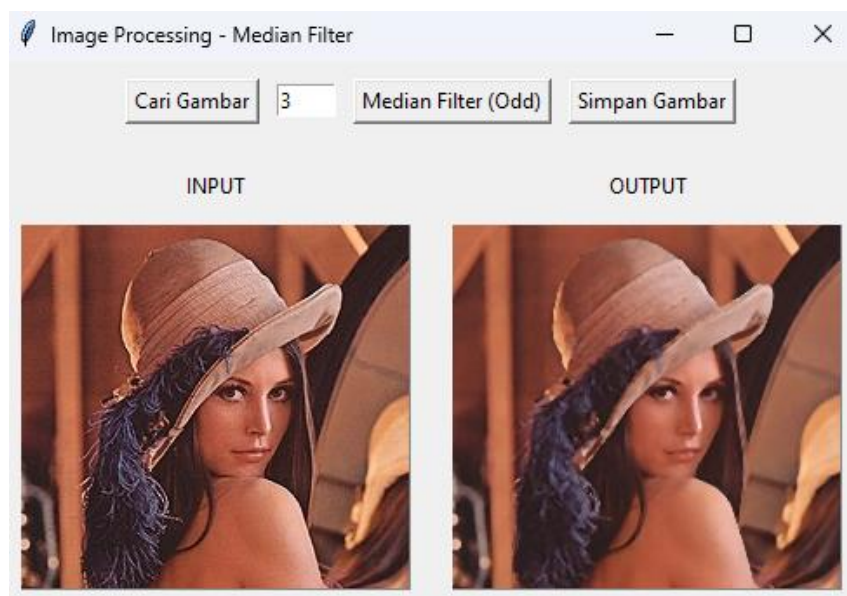
2.1 Menampilkan Input dan output

Untuk mengaplikasikan lebih lanjut maka input dan output akan ditampilkan. Input adalah gambar yang belum diproses dan Output adalah gambar yang telah diproses seperti yang tampak pada gambar 31 dan 32. Gambar 31 merupakan tampilan sebelum gambar diinput dan gambar 32 merupakan tampilan setelah gambar diinput dan diberikan proses yakni median filter kerne 3×3 dan hasilnya ditampilkan kembali. Untuk bermain di GUI

pengolahan citra perlu memahami pengolahan citra pada python terlebih dahulu [8].



Gambar 31. Layout pengolahan citra sederhana (sebelum proses)



Gambar 32. Layout pengolahan citra sederhana (setelah diproses)

Library yang dibutuhkan dipanggil yakni tkinter, PIL, cv2 dan numpy. Lalu format gambar diatur, kemudian beberapa fungsi dibangun. Ada fungsi untuk input gambar, fungsing untuk menampilkan hasil (output), fungsi untuk median filter, fungsi untuk menyimpan gambar. Kernel median filter yang diinput haruslah angka-angka ganjil. Lalu dilanjutkan dengan pemberian nama menu yakni: Image Processing - Median Filter. Tombol-tombol yang dibutuhkan yakni Simpan Gambar, Cari Gambar, dan Median Filter (odd). Diatas input dan output gambar diberikan caption INPUT dan OUTPUT.

Untuk menampilkan gambar 31 dan 32 maka dapat dijalankan program di bawah ini:

#Memanggil Library

```
import tkinter as tk
from tkinter import filedialog, messagebox
from PIL import Image, ImageTk
import cv2
import numpy as np
```

#Membangun Fungsi untuk Format Gambar

```
def open_image():
    global img, img_display_input
    file_path = filedialog.askopenfilename(
        filetypes=[("Image files", "*.jpg *.jpeg *.png *.bmp *.tiff")]
    )
    if file_path:
        img = cv2.imread(file_path)
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        display_input_image(img)
```

#Membangun Fungsi Input Gambar

```
def display_input_image(image):
    global img_display_input
    image = Image.fromarray(image)
    img_display_input = ImageTk.PhotoImage(image)
    canvas_input.delete("all")
    canvas_input.create_image(0, 0, anchor=tk.NW, image=img_display_input)
```

#Membangun Fungsi Output Gambar

```
def display_output_image(image):
```

```

global img_display_output
image = Image.fromarray(image)
img_display_output = ImageTk.PhotoImage(image)
canvas_output.delete("all")
canvas_output.create_image(0, 0, anchor=tk.NW, image=img_display_output)

```

#Membangun Fungsi Median Filter

```

def apply_median_filter():
    global img
    if img is None:
        messagebox.showerror("Error", "No image loaded")
        return

    try:
        ksize = int(entry_kernel_size.get())
        if ksize % 2 == 0 or ksize <= 0:
            raise ValueError("Kernel size must be a positive odd number")

        filtered_img = cv2.medianBlur(img, ksize)
        display_output_image(filtered_img)
    except ValueError as e:
        messagebox.showerror("Error", str(e))

```

#Membangun Fungsi menyimpan gambar

```

def save_image():
    global img_display_output
    if img_display_output is None:
        messagebox.showerror("Error", "No image to save")
        return

    file_path = filedialog.asksaveasfilename(
        defaultextension=".png",
        filetypes=[("PNG files", "*.png"), ("JPEG files", "*.jpg"), ("All files", "*.*")]
    )
    if file_path:
        saved_img = cv2.cvtColor(np.array(img_display_output), cv2.COLOR_RGB2BGR)
        cv2.imwrite(file_path, saved_img)

```

#Memberi Nama Menu

```

root = tk.Tk()
root.title("Image Processing - Median Filter")

```

#Inisialisasi Variabel

```
global img, img_display_input, img_display_output
img = None
img_display_input = None
img_display_output = None
```

#GUI Layout

```
frame_controls = tk.Frame(root)
frame_controls.pack(side=tk.TOP, pady=10)
```

#Mempersiapkan button

```
btn_open = tk.Button(frame_controls, text="Cari Gambar", command=open_image)
btn_open.pack(side=tk.LEFT, padx=5)
```

#Mempersiapkan kernel median filter

```
entry_kernel_size = tk.Entry(frame_controls, width=5)
entry_kernel_size.insert(0, "3")
entry_kernel_size.pack(side=tk.LEFT, padx=5)
```

```
btn_apply = tk.Button(frame_controls, text="Median Filter (Odd)",
command=apply_median_filter)
btn_apply.pack(side=tk.LEFT, padx=5)
```

#Mempersiapkan tombol button

```
btn_save = tk.Button(frame_controls, text="Simpan Gambar",
command=save_image)
btn_save.pack(side=tk.LEFT, padx=5)
```

```
frame_images = tk.Frame(root)
frame_images.pack(side=tk.TOP, pady=10)
```

```
canvas_input_label = tk.Label(frame_images, text="INPUT")
canvas_input_label.grid(row=0, column=0, padx=10, pady=5)
```

```
canvas_output_label = tk.Label(frame_images, text="OUTPUT")
canvas_output_label.grid(row=0, column=1, padx=10, pady=5)
```

```
canvas_input = tk.Canvas(frame_images, width=224, height=210, bg="gray")
canvas_input.grid(row=1, column=0, padx=10, pady=5)
```

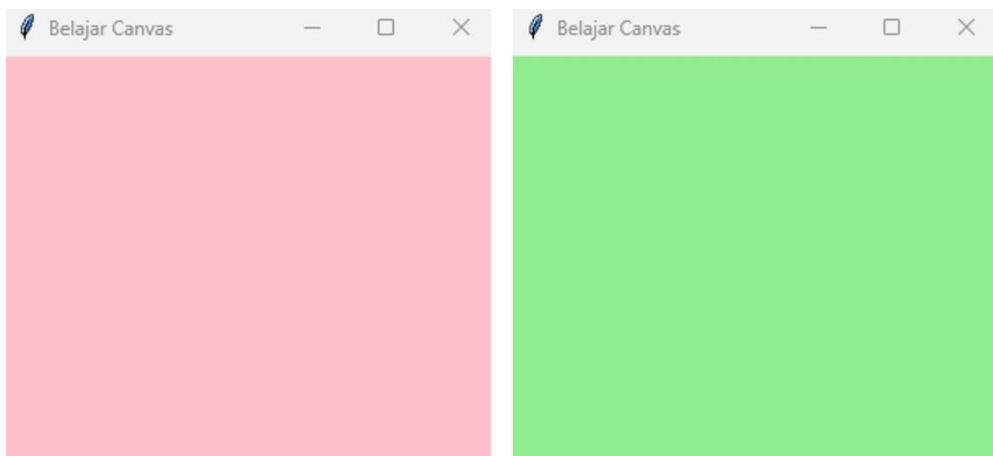
```
canvas_output = tk.Canvas(frame_images, width=224, height=210, bg="gray")
canvas_output.grid(row=1, column=1, padx=10, pady=5)
```

#Mengakhiri Program

```
root.mainloop()
```

2.2 Canvas

Pada sub bab sebelumnya, fungsi canvas dibutuhkan untuk menampilkan gambar, dan pada sub ini akan dijelaskan dengan lebih detail penggunaan canvas. Untuk mulai menggunakan canvas pada tkinter maka kita mulai menentukan warna background dan resolusi menunya seperti yang tampak pada gambar 33 di bawah ini [7-8].



Gambar 33. Background Canvas

Adapun untuk menampilkan gambar 33 di atas dapat menjalankan program di bawah ini:

```
#Memanggil Library
from tkinter import *
root = Tk()

#Memberikan Judul
root.title("Belajar Canvas")
```

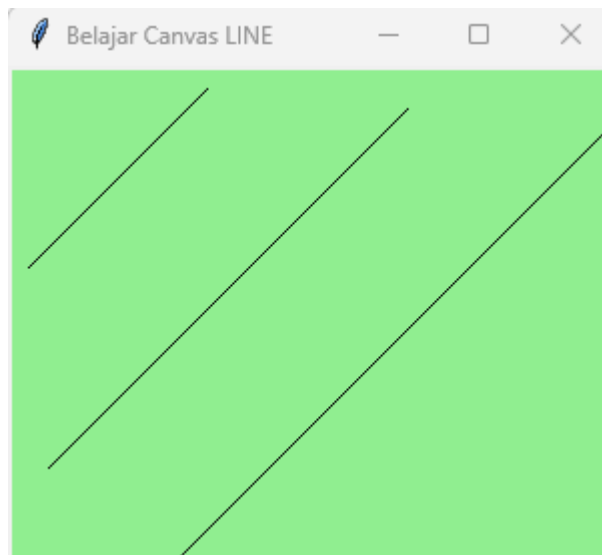
```
#Menentukan background dan ukuran
```

```
C = Canvas(root, bg="light green", height=250, width=300)  
C.pack()
```

```
#Mengakhiri Program
```

```
root.mainloop()
```

Lalu beberapa garis akan ditambahkan ke dalam background seperti yang tampak pada gambar 34 di bawah ini:



Gambar 34. Garis pada Canvas

Kodingan di bawah ini dapat menghasilkan tampilan seperti gambar 34 di atas.

```
#Memanggil Library
```

```
from tkinter import *  
root = Tk()
```

```
#Memberikan Judul
```

```
root.title("Belajar Canvas LINE")
```

```
#Menentukan background dan ukuran
```

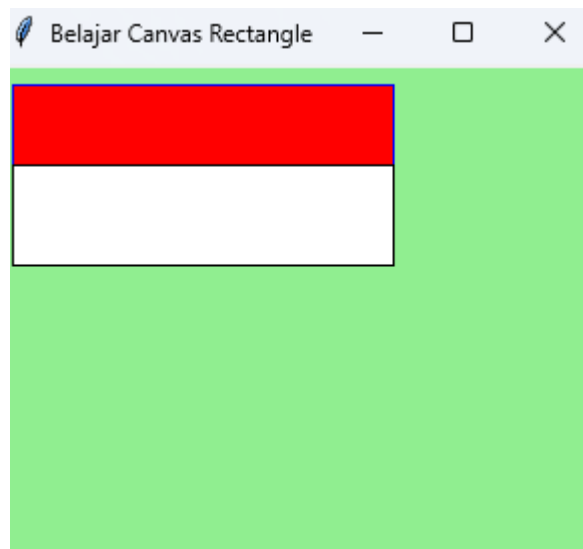
```
C = Canvas(root, bg="light green", height=250, width=300)
line1 = C.create_line(10, 100, 100, 10, fill="black")
line2 = C.create_line(20, 200, 200, 20, fill="black")
line3 = C.create_line(30, 300, 300, 30, fill="black")
```

#Mengakhiri Program

```
C.pack()
root.mainloop()
```

Lanjutan dari kodingan sebelumnya, maka 3 garis ditambahkan menggunakan fungsi `create_line` dan sumbu-sumbunya dimasukkan yakni (x_0, y_0) dan (x_1, y_1) lalu silahkan memasukkan warna yang disukai dan pada kodingan ini, penulis memilih warna black.

Setelah menampilkan garis lalu lanjut untuk menampilkan persegi panjang (rectangle) seperti yang tampak pada gambar 35 di bawah ini:



Gambar 35. Rectangle pada Canvas

Gambar 35 di atas dapat dihasilkan melalui pemrograman di bawah ini:

```
#Memanggil Library
from tkinter import *
root = Tk()
```

```
#Memberikan Judul
```

```
root.title("Belajar Canvas Rectangle")
```

```
#Menentukan rectangle 1 dan 2
```

```
C = Canvas(root, bg="light green", height=250, width=300)
```

```
rectangle1= C.create_rectangle(10, 10, 200, 50, fill='red', outline='blue')
```

```
rectangle2= C.create_rectangle(10, 100, 200, 50, fill='white', outline='black')
```

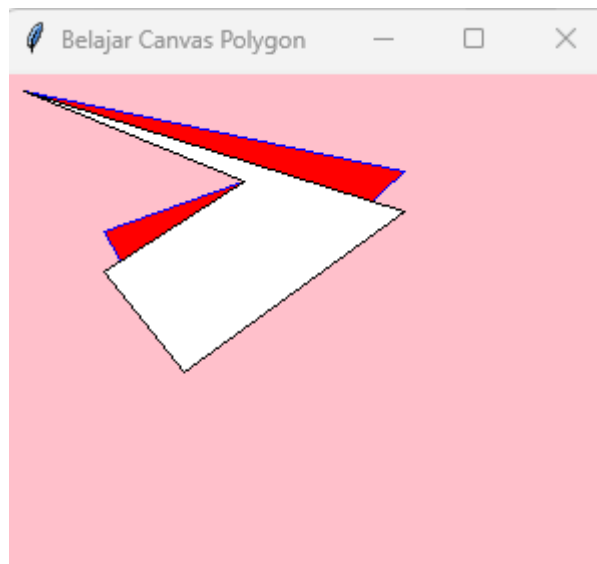
```
#Mengakhiri Program
```

```
C.pack()
```

```
root.mainloop()
```

Pada kodingan ini maka untuk memberikan rectangle digunakan fungsi `create_rectangle` dan diberikan koordinatnya. Lalu lanjut dengan memberikan warna pada isi dan outline rectangle.

Polygon juga bisa ditampilkan pada canvas seperti yang tampak pada gambar 36 di bawah ini. Gambar 36 menampilkan dua polygon.



Gambar 36. Polygon pada Canvas

Adapun gambar 36 dapat dihasilkan melalui pemrograman di bawah ini:

```
#Memanggil Library
```

```
from tkinter import *
```

```
root = Tk()
```

```
#Memberikan Judul
```

```
root.title("Belajar Canvas Polygon")
```

```
#Menentukan Polygon 1 dan 2
```

```
C = Canvas(root, bg="pink", height=250, width=300)
```

```
polygon1= C.create_polygon(10, 10, 200, 50, 90, 150, 50, 80, 120, 55, fill='red',  
outline='blue')
```

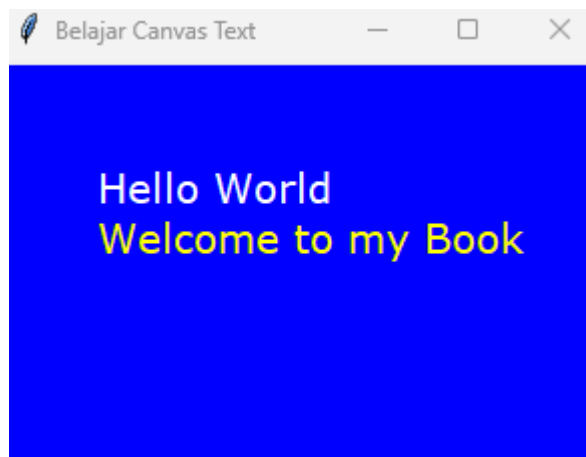
```
polygon2= C.create_polygon(10, 10, 200, 70, 90, 150, 50, 100, 120, 55, fill='white',  
outline='black')
```

```
#Mengakhiri Program
```

```
C.pack()
```

```
root.mainloop()
```

Pada kodingan ini dipanggil library yang diperlukan dan mempersiapkan root window. Kemudian diberikan penamaan menu dengan fungsi root.title. Lalu, dipersiapkan resolusi canvas dan background. Kemudian polygon 1 dan 2 dibangun dengan titik koordinat masing-masing, warna isi dan outlinenya. Memberikan text pada canvas dapat juga dilakukan seperti yang tampak pada gambar 37 di bawah ini:



Gambar 37. Text pada Canvas

Gambar 37 di atas dapat diperoleh melalui kodingan di bawah ini:

```
#Memanggil Library
```

```
from tkinter import *
```

```
root = Tk()
```

#Memberikan Judul

```
root.title("Belajar Canvas Text")
```

#Menentukan Text

```
C = Canvas(root, bg="blue", width=300, height=300)
```

```
D= C.create_text(50, 50, text='Hello World',anchor='nw', fill='white', font=('verdana', 15))
```

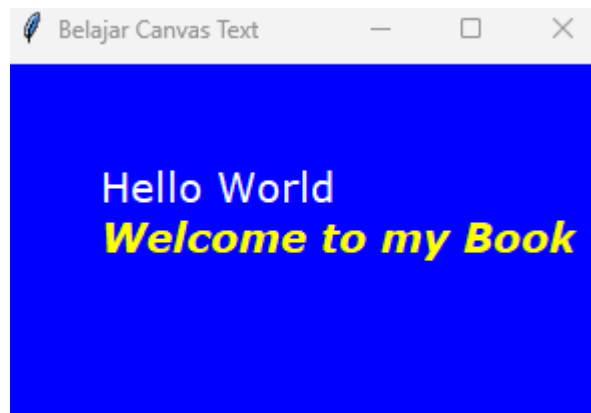
```
D2= C.create_text(50, 75, text='Welcome to my Book',anchor='nw', fill='yellow', font=('verdana', 15))
```

#Mengakhiri Program

```
C.pack()
```

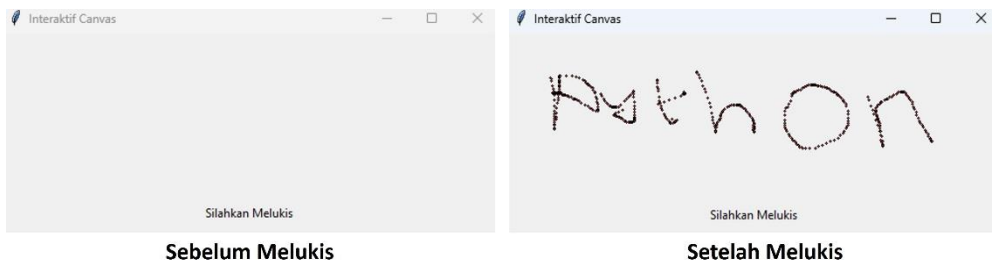
```
root.mainloop()
```

Adapun kodingan ini hamper menyerupai kodingan sebelumnya pada canvas. Fungsi yang digunakan adalah ***create_text*** dan di dalamnya ada beberapa koordinat yang harus diisi, lalu mengisi text yang akan ditampilkan, menentukan anchor dan jenis font yang diinginkan. Untuk font, bisa dilakukan editing seperti ini: ***font=('verdana 15 bold italic')*** sehingga tampilannya tampak pada gambar 38 di bawah ini.



Gambar 38. Text pada canvas dengan beda format

Kemudian dengan canvas maka kegiatan interaktif bisa juga dilangsungkan seperti yang tampak pada gambar 39 di bawah ini:



Gambar 39. Interaktif Canvas

Adapun pemrograman untuk gambar 39 dapat menjalankan di bawah ini:

#Memanggil Library

```
from tkinter import *
```

```
#Mempersiapkan ukuran canvas
```

```
canvas_width = 500
```

```
canvas_height = 150
```

#Fungsi

```
def paint( event ):
```

```
    python_green = "#D32F2F"
```

```
    x1, y1 = ( event.x - 1 ), ( event.y - 1 )
```

```
    x2, y2 = ( event.x + 1 ), ( event.y + 1 )
```

```
    w.create_oval( x1, y1, x2, y2, fill = python_green )
```

#Mempersiapkan window

```
master = Tk()
```

```
master.title( "Interaktif Canvas" )
```

```
w = Canvas(master,
```

```
            width=canvas_width,
```

```
            height=canvas_height)
```

```
w.pack(expand = YES, fill = BOTH)
```

```
w.bind( "<B1-Motion>", paint )
```

#Caption Menu

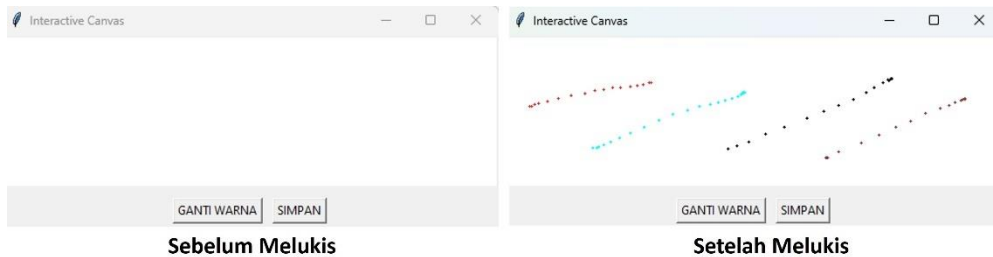
```
message = Label( master, text = "Silahkan Melukis", height= 3, width=15 )
```

```
message.pack( side = BOTTOM )
```

mainloop()

Pada kodingan diawal, library akan dipanggil dan ukuran canvas ditentukan yakni 500 x 150. Kemudian fungsi untuk interaktif dipersiapkan: `def paint(event)`. Di dalam fungsi berisikan lokasi untuk interaktif. Kemudian, root window diinput ke dalam variable master dan judul menu diberikan: Interaktif Canvas. Pesan akhir: Silahkan Melukis ditempatkan ditengah bawah dan kemudian program selesai.

Dari interaktif canvas, bisa dilanjutkan untuk modifikasi pilihan warna dan setelah melakukan interaksi maka hasil interaksi dapat disimpan dalam format PNG. Adapun tampilan interaktifnya dapat dilihat pada gambar 40 berikut ini.



Gambar 40. Interaktif Canvas dengan pilihan warna

Adapun gambar 40 tersebut dapat dijalankan melalui program di bawah ini:

#Memanggil Library

```
from tkinter import *
from tkinter import colorchooser, filedialog
```

#Mempersiapkan Ukuran Canvas

```
canvas_width = 500
canvas_height = 150
current_color = "#D32F2F" # Default brush color
```

#Mempersiapkan Fungsi Untuk Menggambar

```
def paint(event):
    x1, y1 = (event.x - 1), (event.y - 1)
    x2, y2 = (event.x + 1), (event.y + 1)
    canvas.create_oval(x1, y1, x2, y2, fill=current_color, outline=current_color)
```

#Mempersiapkan Fungsi Untuk mengganti warna

```
def change_color():
    global current_color
    color_code = colorchooser.askcolor(title="Choose Brush Color")[1]
    if color_code:
        current_color = color_code
```

#Mempersiapkan Fungsi Untuk menyimpan gambar

```
def save_drawing():
    file_path = filedialog.asksaveasfilename(defaultextension=".png",
                                             filetypes=[("PNG files", "*.png"), ("All files", "*.*")])
    if file_path:
        # Save the canvas content as an image
        canvas.postscript(file=file_path + ".eps") # Save as EPS file
        try:
            from PIL import Image
```

```

        img = Image.open(file_path + ".eps")
        img.save(file_path)
    finally:
        import os
        os.remove(file_path + ".eps")

#Mempersiapkan Window
master = Tk()
master.title("Interactive Canvas")

#Mempersiapkan Layout Canvas
canvas = Canvas(master, width=canvas_width, height=canvas_height, bg="white")
canvas.pack(expand=YES, fill=BOTH)

#Mempersiapakn Fungsi Mouse Kiri
canvas.bind("<B1-Motion>", paint)

#Mempersiapkan Button
frame_controls = Frame(master)
frame_controls.pack(side=BOTTOM, pady=10)

btn_color = Button(frame_controls, text="GANTI WARNA", command=change_color)
btn_color.pack(side=LEFT, padx=5)

btn_save = Button(frame_controls, text="SIMPAN", command=save_drawing)
btn_save.pack(side=LEFT, padx=5)

#Menyelesaikan Program
mainloop()

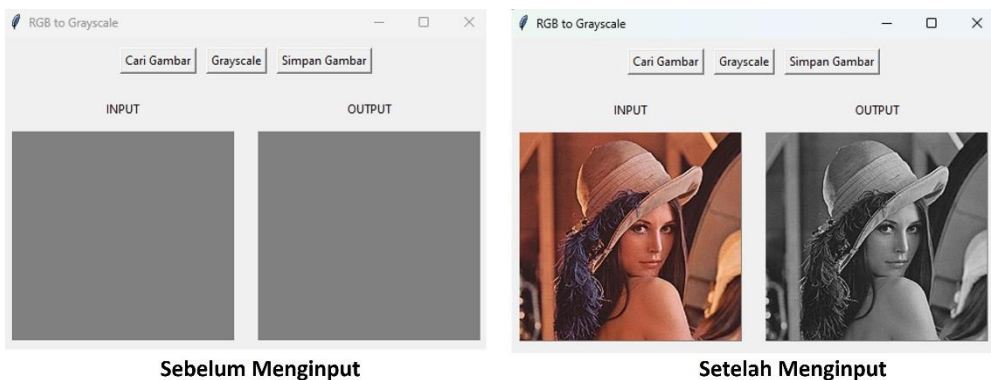
```

Kodingan untuk gambar 40 diawali dengan memanggil library yang dibutuhkan seperti tkinter, colorchooser, dan filedialog. Ukuran canvas yang ditentukan yakni 500 x 150. Fungsi untuk menggambar dipersiapkan dengan def paint(event) dan fungsi mengganti warna dengan def change_color. Fungsi global memanggil current_color yang sudah didefinisikan diawal. Untuk menyimpan gambar tersendiri ada fungsi yang disediakan yakni def save_drawing yang menyimpan file dalam format PNG. Menu diberi nama Interactive Canvas dan fungsi mouse kiri diaktifkan dengan B1-Motion. Ada

beberapa button yang dipersiapkan seperti Ganti Warna, Simpan dan program selesai.

2.3 GUI: RGB to Grayscale

Setelah membahas canvas di bab sebelumnya, maka saat ini lanjut mengaplikasikan GUI pada pengolahan citra digital [6]. Adapun Langkah selanjutnya adalah menampilkan konversi RGB to Grayscale.



Gambar 41. GUI RGB to Grayscale

#Memanggil Library

```
import tkinter as tk
from tkinter import filedialog, messagebox, Label
from PIL import Image, ImageTk
import cv2
import numpy as np
```

#Fungsi membuka gambar

```
def open_image():
    global img, img_display_input
    file_path = filedialog.askopenfilename(
        filetypes=[("Image files", "*.jpg *.jpeg *.png *.bmp *.tiff")]
    )
    if file_path:
        img = cv2.imread(file_path)
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
```

```
display_input_image(img)
```

#Fungsi Menampilkan Gambar Input

```
def display_input_image(image):  
    global img_display_input  
    image = Image.fromarray(image)  
    img_display_input = ImageTk.PhotoImage(image)  
    canvas_input.delete("all")  
    canvas_input.create_image(0, 0, anchor=tk.NW, image=img_display_input)
```

#Fungsi Menampilkan Gambar Output

```
def display_output_image(image):  
    global img_display_output  
    image = Image.fromarray(image)  
    img_display_output = ImageTk.PhotoImage(image)  
    canvas_output.delete("all")  
    canvas_output.create_image(0, 0, anchor=tk.NW, image=img_display_output)
```

#Fungsi Filter Grayscale

```
def grayscale():  
    global img, img_display_output  
    if img is None:  
        messagebox.showerror("Error", "No image loaded")  
        return  
  
    gray_img = cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)  
    gray_img_rgb = cv2.cvtColor(gray_img, cv2.COLOR_GRAY2RGB) # Convert back to  
    RGB for consistent display  
    display_output_image(gray_img_rgb)
```

#Fungsi Untuk Menyimpan Gambar

```
def save_image():  
    global img_display_output  
    if img_display_output is None:  
        messagebox.showerror("Error", "No image to save")  
        return  
  
    file_path = filedialog.asksaveasfilename(  
        defaultextension=".png",  
        filetypes=[("PNG files", "*.png"), ("JPEG files", "*.jpg"), ("All files", "*.*)"]  
    )  
    if file_path:  
        saved_img = cv2.cvtColor(np.array(img_display_output), cv2.COLOR_RGB2BGR)
```

```

cv2.imwrite(file_path, saved_img)

#Inisialisasi GUI
def main():
    global root, canvas_input, canvas_output, img, img_display_input,
img_display_output

    root = tk.Tk()
    root.title("RGB to Grayscale")

    #Inisialisasi Variable
    img = None
    img_display_input = None
    img_display_output = None

    #Layout GUI
    frame_controls = tk.Frame(root)
    frame_controls.pack(side=tk.TOP, pady=10)

    btn_open = tk.Button(frame_controls, text="Cari Gambar",
command=open_image)
    btn_open.pack(side=tk.LEFT, padx=5)

    btn_apply = tk.Button(frame_controls, text="Grayscale", command=grayscale)
    btn_apply.pack(side=tk.LEFT, padx=5)

    btn_save = tk.Button(frame_controls, text="Simpan Gambar",
command=save_image)
    btn_save.pack(side=tk.LEFT, padx=5)

    frame_images = tk.Frame(root)
    frame_images.pack(side=tk.TOP, pady=10)

    canvas_input_label = tk.Label(frame_images, text="INPUT")
    canvas_input_label.grid(row=0, column=0, padx=10, pady=5)

    canvas_output_label = tk.Label(frame_images, text="OUTPUT")
    canvas_output_label.grid(row=0, column=1, padx=10, pady=5)

    canvas_input = tk.Canvas(frame_images, width=224, height=210, bg="gray")
    canvas_input.grid(row=1, column=0, padx=10, pady=5)

    canvas_output = tk.Canvas(frame_images, width=224, height=210, bg="gray")

```

```
canvas_output.grid(row=1, column=1, padx=10, pady=5)
```

```
#Menyelesaikan Program
```

```
root.mainloop()
```

```
if __name__ == "__main__":  
    main()
```

Pada gambar 41 dengan kodingan yang sudah dijabarkan sebelumnya, dapat dilihat beberapa library yang dibutuhkan dipanggil seperti tkinter, filedialog, messagebox, Label, PIL, ImageTk, Cv2 dan NumPy. Lalu fungsi membuka gambar dipersiapkan yakni def open_image() lalu lanjut dengan menggunakan fungsi global dengan dua input yakni img dan img_display_input. Gambar yang dibuka dengan beberapa jenis file yakni jpg, jpeg, png, bmp dan tiff. Lalu gambar menampilkan input dengan fungsi def display_input_image yang inputannya adalah image. Fungsi menampilkan gambar dipersiapkan dengan fungsi def display_output_image(image) dan ada fungsi global img_display_output yang dimana fungsi global ini juga ada di fungsi Grayscale dan fungsi menyimpan gambar. Setelah menyelesaikan beberapa fungsi penting, lalu lanjut dengan inisialisasi GUI, variable input dan output, mempersiapkan tombol Cari Gambar, Grayscale, Simpan Gambar dan program selesai.

2.4 Menampilkan Gambar Tanpa Fungsi

Pada bab 2.1 sudah dibahas bagaimana menampilkan gambar pada GUI dengan menggunakan fungsi. Di sub bab ini ada acara lebih sederhana untuk menampilkan gambar seperti yang tampak pada gambar 42 di bawah ini:



Gambar 42. Menampilkan Gambar

Adapun gambar 42 dapat diperoleh melalui kodingan di bawah ini:

#Memanggil Library

```
import tkinter as tk
from PIL import Image, ImageTk
```

#Membuat Main Window

```
root = tk.Tk()
root.title("Menampilkan Gambar")
```

#Mempersiapkan Ukuran Window

```
root.geometry("300x300")
```

#Loading Gambar

```
image = Image.open("LENA.jpg")
A = ImageTk.PhotoImage(image)
```

#Mempersiapkan Label

```
label = tk.Label(root, image=A)
label.pack()
```

#Menyelesaikan program

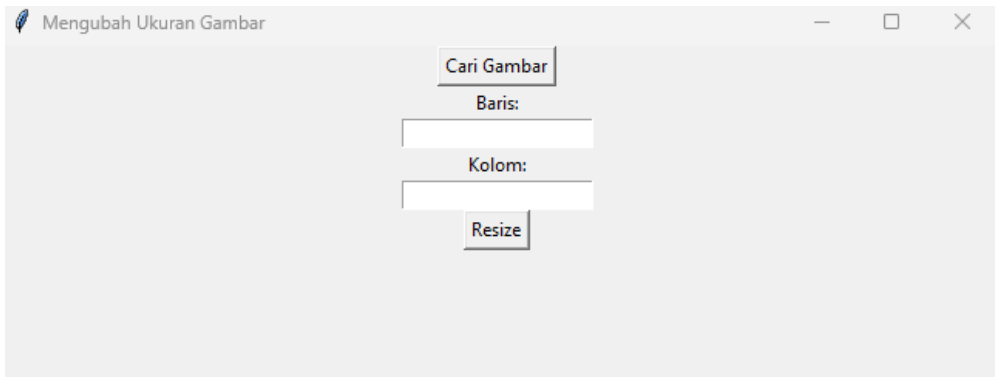
```
root.mainloop()
```

Untuk menampilkan gambar 42, kodingan diawali dengan menghadirkan library yang dibutuhkan seperti tkinter, PIL, Image dan ImageTk. Lalu main window dipersiapkan dengan judul Menampilkan Gambar. Besaran window

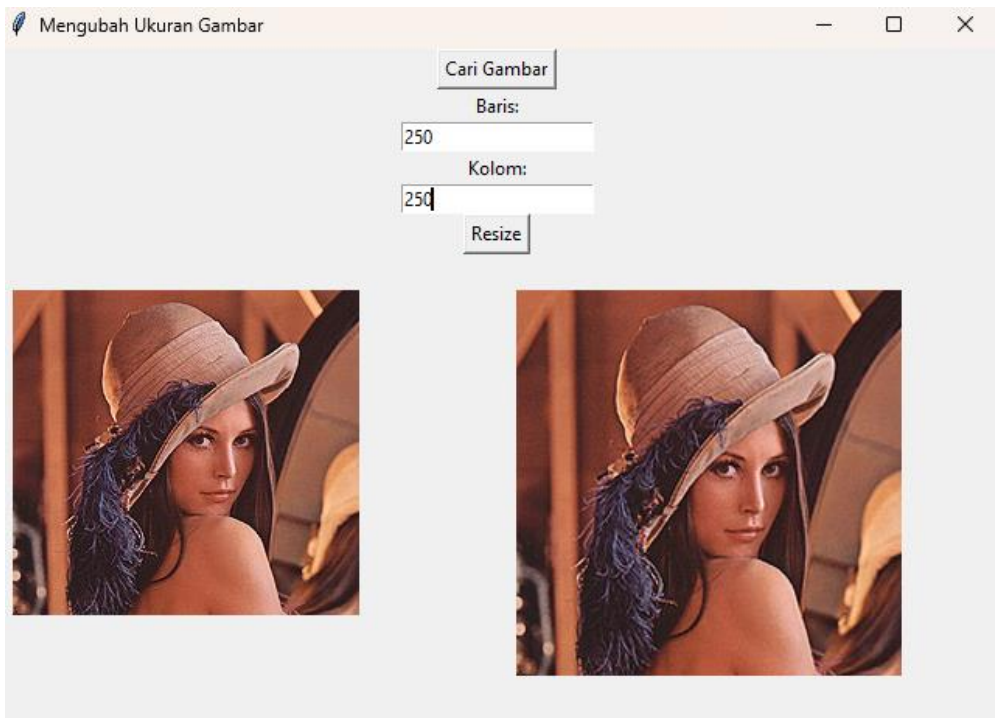
diatur 300×300 . Lalu gambar yang akan ditampilkan diisi di `Image.Open`. Gambar yang akan sering digunakan diletakkan pada direktori yang sama dengan file python yang dijalankan. Lalu label dipersiapkan dengan root dan gambar kemudian program selesai.

2.5 Mengubah Resolusi

Aplikasi selanjutnya dari GUI yang bisa diterapkan pada pengolahan citra adalah mengubah resolusi citra seperti yang tampak pada gambar 43 dan 44 di bawah ini:



Gambar 43. Layout Resize Image



Gambar 44. Input dan Output pada Layout

Adapun gambar 43 dan 44 dapat dijalankan dengan program di bawah ini:

#Memanggil Library

```
import tkinter as tk
from tkinter import filedialog
from PIL import Image, ImageTk
```

#Fungsi Membuka Citra

```
def open_image():
    # Open a file dialog to select an image file
    file_path = filedialog.askopenfilename()
    if file_path:
        # Load the image
        global original_image
        original_image = Image.open(file_path)

        # Display the original image
        display_image(original_image, canvas_original)
```

#Fungsi Mengubah Resolusi

```
def resize_image():
    if original_image:
        try:
            # Get the width and height from the input fields
            width = int(width_entry.get())
            height = int(height_entry.get())

            # Resize the image based on the input dimensions
            resized_image = original_image.resize((width, height))

            # Display the resized image
            display_image(resized_image, canvas_resized)
        except ValueError:
            # Handle invalid input
            error_label.config(text="Please enter valid integers for width and height.")
            return
```

#Fungsi Menampilkan Gambar

```
def display_image(img, canvas):
    # Convert the PIL image to a Tkinter-compatible image
    img_tk = ImageTk.PhotoImage(img)

    # Create or update the canvas with the image
```

```
canvas.create_image(0, 0, anchor="nw", image=img_tk)
canvas.image = img_tk # Keep a reference to the image to prevent garbage
collection
```

#Mempersiapkan Main Window

```
root = tk.Tk()
root.title("Mengubah Ukuran Gambar")
```

#Global Variabel untuk menyimpan gambar

```
original_image = None
```

#Button Untuk Open Image

```
open_button = tk.Button(root, text="Cari Gambar", command=open_image)
open_button.pack()
```

#Mempersiapkan Input Baris

```
width_label = tk.Label(root, text="Baris:")
width_label.pack()
```

```
width_entry = tk.Entry(root)
width_entry.pack()
```

#Mempersiapkan Input Kolom

```
height_label = tk.Label(root, text="Kolom:")
height_label.pack()
```

```
height_entry = tk.Entry(root)
height_entry.pack()
```

#Button Untuk Resize

```
resize_button = tk.Button(root, text="Resize", command=resize_image)
resize_button.pack()
```

#Label Untuk Pesan Error

```
error_label = tk.Label(root, text="", fg="red")
error_label.pack()
```

#Canvas Untuk Menampilkan Gambar Input

```
canvas_original = tk.Canvas(root, width=300, height=300)
canvas_original.pack(side="left", padx=10)
```

#Canvas Untuk Menampilkan Gambar Output

```
canvas_resized = tk.Canvas(root, width=300, height=300)
```

```
canvas_resized.pack(side="right", padx=10)
```

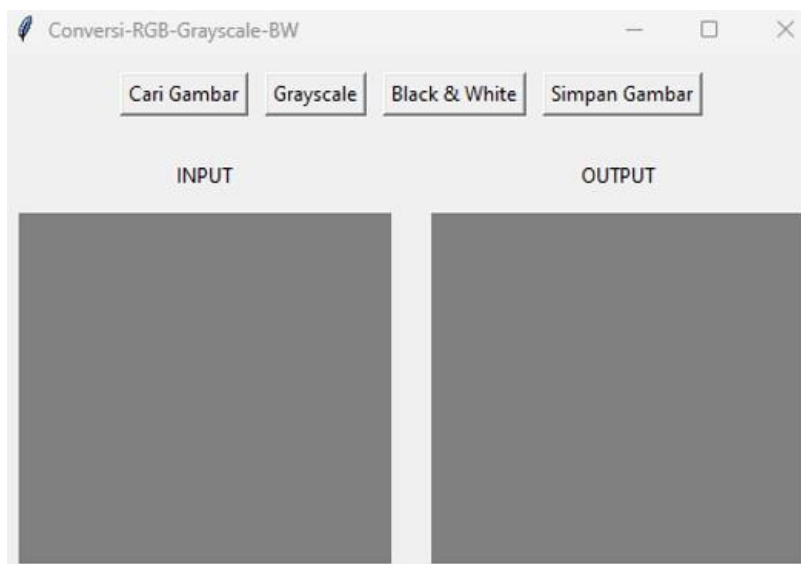
```
#Menyelesaikan Program
```

```
root.mainloop()
```

Program diawali dengan memanggil library yang dibutuhkan seperti tkinter, filedialog, PIL, Image dan ImageTk. Kemudian beberapa fungsi dibangun seperti fungsi Membuka Gambar, Mengubah Resolusi, Menampilkan Gambar, dan Mempersiapkan Main window. Global Variabel dalam variable `original_image` untuk menyimpan gambar. Lalu beberapa button dipersiapkan seperti Cari Gambar, Baris, Kolom, Resize dan program selesai.

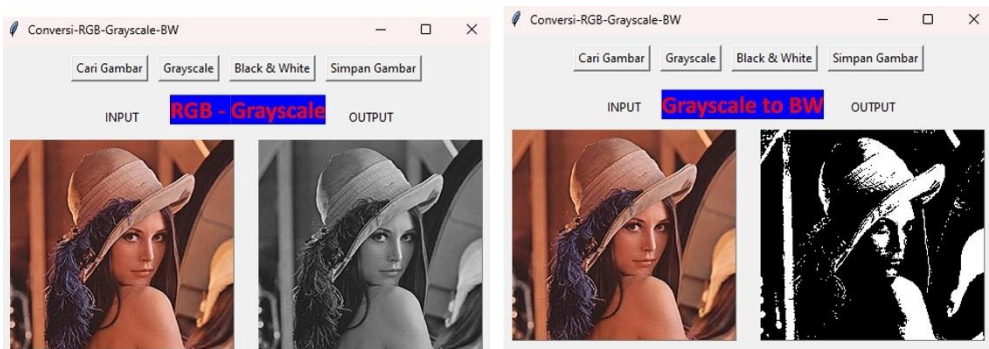
2.6 GUI: RGB to Grayscale to Black and White (BW)

Selanjutnya, GUI yang akan dibangun dalam pengolahan citra adalah lanjutan dari sub bab 2.3 yakni menambahkan menu black and white (BW) [9-12]. Tampilan layout seperti yang terlihat di gambar 45 di bawah ini:



Gambar 45. Layout Conversi RGB to Grayscale to BW

Kemudian tampilan setelah gambar dimasukkan dan diproses adalah seperti yang tampak pada gambar 46 berikut ini [13]:



Gambar 46. Setelah Input Gambar

Adapun tampilan 45 dan 46 dapat diperoleh melalui pemrograman di bawah ini:

#Memanggil Library

```
import tkinter as tk
from tkinter import filedialog, messagebox, Label
from PIL import Image, ImageTk
import cv2
import numpy as np
```

#Fungsi membuka gambar

```
def open_image():
    global img, img_display_input
    file_path = filedialog.askopenfilename(
        filetypes=[("Image files", "*.jpg *.jpeg *.png *.bmp *.tiff")]
    )
    if file_path:
        img = cv2.imread(file_path)
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        display_input_image(img)
```

#Fungsi Menampilkan Gambar Input

```
def display_input_image(image):
    global img_display_input
    image = Image.fromarray(image)
    img_display_input = ImageTk.PhotoImage(image)
    canvas_input.delete("all")
    canvas_input.create_image(0, 0, anchor=tk.NW, image=img_display_input)
```

#Fungsi Menampilkan Gambar Output

```
def display_output_image(image):
    global img_display_output
    image = Image.fromarray(image)
    img_display_output = ImageTk.PhotoImage(image)
    canvas_output.delete("all")
    canvas_output.create_image(0, 0, anchor=tk.NW, image=img_display_output)
```

#Fungsi Filter Grayscale

```
def grayscale():
    global img, img_display_output
    if img is None:
        messagebox.showerror("Error", "No image loaded")
        return

    gray_img = cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)
    gray_img_rgb = cv2.cvtColor(gray_img, cv2.COLOR_GRAY2RGB) # Convert back to
    RGB for consistent display
    display_output_image(gray_img_rgb)
```

#Fungsi Filter Black and White

```
def black_and_white():
    global img, img_display_output
    if img is None:
        messagebox.showerror("Error", "No image loaded")
        return

    gray_img = cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)
    _, bw_img = cv2.threshold(gray_img, 127, 255, cv2.THRESH_BINARY) # Apply
    thresholding
    bw_img_rgb = cv2.cvtColor(bw_img, cv2.COLOR_GRAY2RGB) # Convert back to
    RGB for consistent display
    display_output_image(bw_img_rgb)
```

#Fungsi Untuk Menyimpan Gambar

```
def save_image():
```

```

global img_display_output
if img_display_output is None:
    messagebox.showerror("Error", "No image to save")
    return

file_path = filedialog.asksaveasfilename(
    defaultextension=".png",
    filetypes=[("PNG files", "*.png"), ("JPEG files", "*.jpg"), ("All files", "*.*")]
)
if file_path:
    saved_img = cv2.cvtColor(np.array(img_display_output), cv2.COLOR_RGB2BGR)
    cv2.imwrite(file_path, saved_img)

```

#Inisialisasi GUI

```

def main():
    global root, canvas_input, canvas_output, img, img_display_input, img_display_output

    root = tk.Tk()
    root.title("Conversi-RGB-Grayscale-BW")

    #Inisialisasi Variable
    img = None
    img_display_input = None
    img_display_output = None

    #Layout GUI
    frame_controls = tk.Frame(root)
    frame_controls.pack(side=tk.TOP, pady=10)

    btn_open = tk.Button(frame_controls, text="Cari Gambar",
command=open_image)
    btn_open.pack(side=tk.LEFT, padx=5)

    btn_grayscale = tk.Button(frame_controls, text="Grayscale", command=grayscale)
    btn_grayscale.pack(side=tk.LEFT, padx=5)

    btn_bw = tk.Button(frame_controls, text="Black & White",
command=black_and_white)
    btn_bw.pack(side=tk.LEFT, padx=5)

```

```

    btn_save = tk.Button(frame_controls, text="Simpan Gambar",
command=save_image)
    btn_save.pack(side=tk.LEFT, padx=5)

    frame_images = tk.Frame(root)
    frame_images.pack(side=tk.TOP, pady=10)

    canvas_input_label = tk.Label(frame_images, text="INPUT")
    canvas_input_label.grid(row=0, column=0, padx=10, pady=5)

    canvas_output_label = tk.Label(frame_images, text="OUTPUT")
    canvas_output_label.grid(row=0, column=1, padx=10, pady=5)

    canvas_input = tk.Canvas(frame_images, width=224, height=210, bg="gray")
    canvas_input.grid(row=1, column=0, padx=10, pady=5)

    canvas_output = tk.Canvas(frame_images, width=224, height=210, bg="gray")
    canvas_output.grid(row=1, column=1, padx=10, pady=5)

    #Menyelesaikan Program
    root.mainloop()

```

```

if __name__ == "__main__":
    main()

```

Program diawali dengan memanggil library yang dibutuhkan seperti tkinter, filedialog, messagebox, Label, Open CV, dan NumPy. Ada beberapa fungsi yang dibangun yakni fungsi untuk membuka gambar (def open_image()), fungsi untuk menampilkan input (def display_input_image()), fungsi untuk menampilkan output (def display_output_image()), fungsi untuk grayscale (def grayscale()), fungsi untuk black and white (def black_and_white()), fungsi untuk menyimpan gambar (def save_image()) dan fungsi untuk GUI sendiri. Kemudian beberapa button dipersiapkan seperti Cari Gambar, Grayscale, Black and White, Simpan Gambar. Awal pelaksanaan, user diminta untuk mencari gambar kemudian dilanjutkan dengan menekan tombol grayscale dan akhirnya tombol black and white. Kemudian output citra bisa disimpan dan program selesai.

Daftar Pustaka

1. <https://docs.python.org/3/library/tkinter.html>
2. <https://www.tkinter.com/>
3. <https://tkdocs.com/tutorial/firstexample.html>
4. https://www.tutorialspoint.com/python/python_gui_programming.htm
5. <https://thepythoncode.com/article/make-an-image-editor-in-tkinter-python#creating-the-main-window>.
6. R.C.Gonzalez, R.E.Woods, "*Digital Image Processing 4th Edition*", Pearson, 2018.
7. Bhaskar Chaudhary, "*Tkinter GUI Application Development*", Packt Publishing, Mumbai, 2013.
8. John E.Grayson, "*Python and Tkinter Programming*", Manning, Greenwich, 2000.
9. C.E.Panjaitan, S.W.Tarigan, D.M.Hutagalung, O.Y.Hutajulu, M.D.Mendoza, "*Dasar Pemrograman Basic Python Interface Google Colaboratory*", Unpri Press, Edisi 1, 2024, ISBN: 978-623-8299-28-7.
10. C.E.Panjaitan, D.M.Hutagalung, O.Y.Hutajulu, M.D.Mendoza, "*Pengolahan Citra Digital Menggunakan Python (Digital Image Processing with Python)*", Pena Persana Kerta Utama, Edisi 1, 2025, ISBN: 978-634-204-213-7.
11. C. E.Panjaitan, Y.Panjaitan, D.Sitanggang, S.W.Tarigan, "*Image Processing For Detection of Dengue Virus*", Jurnal Sistem Informasi dan Ilmu Komputer Prima, Vol.7 No.2, February 2024.
12. C. Panjaitan, A. Silaban, M.Napitupulu, J.W.Simatupang, "*Comparison K-Nearest Neighbors (K-NN) and Artificial Neural Network (ANN) in Real Time Entrants Recognition*", IEEE International Seminar on Research of Information and Intelligent System (ISRITI), 2018.
13. Baharuddin, Olnes Yosefa Hutajulu, Muhammad Dominique Mendoza, Hesti Fibriasari, Muhammad Dani Solihin, "*Monograf inovasi teknologi chatbot berbasis artificial intelligence sebagai learning management system*", Pena Persada Kerta Utama, 2023.
14. Marsangkap Silitonga, Olnes Yosefa Hutajulu, Muhammad Dominique Mendoza, "*Pembelajaran Daring Dengan IoT*", Megalitera, 2021.
15. Reni Rahmadani, Muhammad Dominique Mendoza, Olnes Yosefa Hutajulu, Tansa Trisna Astono Putri, Devi Silvia Panjaitan, Azqal Azqia, "*Membuat dan mengelola sumber daya desain graphic design 101*", Pena Persada Kerta Utama, 2022

16. D.M.Hutagalung, C.E.Panjaitan, "*Use of the SAW (Simple Additive Weighting) Decision Making System in Determining the OSIS Board of Senior High Schools*", Journal of Computer Engineering, System and Science, e-ISSN: 2502-714X, 2023

Biografi Penulis

Penulis 1



Christin Erniati Panjaitan merupakan seorang pengajar di prodi Teknik Elektro, Universitas Prima Indonesia. Penulis juga merupakan anggota dari PT PUI Kesehatan Berbasis IoT & Energi Terbarukan. Penulis merupakan lulusan S2 dari National Taiwan University of Science and Technology (NTUST). Penulis selalu menyuarkan ide dan pemikirannya dalam buku, artikel ilmiah skala nasional dan internasional. Sebagai seorang akademisi, penulis juga aktif mengikuti kegiatan dari Kemdikbud seperti Talent Scouting, Peningkatan Kemampuan Bahasa Inggris (PKBI), Sertifikasi Kompetensi, Praktisi Mengajar, Kampus Merdeka, dll. Penulis memiliki ketertarikan di bidang Biomedial Engineering. Untuk berkolaborasi dan menghubungi penulis bisa melalui: christinpanjaitan@unprimdn.ac.id or christin.erniati@gmail.com

Penulis 2



Dini M Hutagalung, dilahirkan di Medan serta menyelesaikan pendidikan Sekolah Dasar dan Sekolah Menengah Bawah dan Atas di Medan. Penulis menyelesaikan pendidikan Sekolah Menengah atas di SMA ST. Thomas Medan tahun 1989. Mengambil S1 Fakultas Pertanian di Universitas Sumatera Utara tahun 1990 dan tamat tahun 1995, kemudian melanjutkan ke jenjang Magister (S2) dengan jurusan Teknik Informasi (Information Technology) di University of East London tahun 2000-2002. Tahun 2013 sampai sekarang , masih aktif mengajar di universitas swasta di kota Medan, dengan pengalaman mengajar sudah 9 tahun. Penulis mempunyai ketertarikan untuk melakukan penelitian di bidang Basis Data, Data Mining dan Digital Image Processing berbasis Python

Penulis aktif melakukan kegiatan Tri Darma dan untuk korespondensi dengan penulis, dapat menghubungi: mhdini@gmail.com.

Penulis 3



Sri Wahyuni Tarigan menempuh pendidikan strata satu di bidang teknik geologi, Institut Teknologi Medan dan menyelesaikan pendidikan pada tahun 1999. Setelah itu bekerja di perusahaan kontraktor dan memulai karir sebagai dosen di Institut Teknologi Medan selama tiga tahun setelah menamatkan pendidikan magister kimia di Universitas Prima Indonesia.. Pada tahun 2014 penulis menjadi staf pengajar di Universitas Prima Indonesia hingga saat ini setelah resign dari perusahaan kontraktor. Berdasarkan latar belakang pendidikan penulis melakukan penelitian di bidang material teknik dan kimia mulai tahun 2019 hingga saat ini kemudian di tuangkan dalam jurnal nasional dan jurnal internasional. Selain jurnal penulis juga menuangkan hasil penelitian dalam buku yang dapat di manfaatkan oleh mahasiswa. Penulis dapat dihubungi di: sriwahyunitarigan@unprimdn.ac.id

Penulis 4



Olnes Yosefa Hutajulu menyelesaikan pendidikan dasar dan menengah di Bandar Huluan dan Pematang Siantar sebelum melanjutkan studi S1 di Pendidikan Teknik Elektro, Universitas Negeri Medan, yang diselesaikan pada tahun 2011. Ia kemudian melanjutkan studi S2 di Universitas Gadjah Mada, Yogyakarta, dan meraih gelar Magister Teknik (M.Eng) pada tahun 2015 dengan dukungan Beasiswa Pendidikan Pascasarjana Dalam Negeri (BPPDN). Pada tahun 2021, ia memperoleh gelar Insinyur (Ir) melalui program Pendidikan Profesi Insinyur di Universitas Negeri Medan. Karier akademiknya dimulai sebagai pengajar di Institut Teknologi Del pada 2016–2017, dilanjutkan di Tanri Abeng University pada 2017–2019, hingga akhirnya menjadi dosen ASN di Universitas Negeri Medan sejak 2019. Sebagai akademisi, ia telah menerbitkan berbagai karya ilmiah bereputasi di tingkat nasional dan internasional, memiliki paten sederhana, serta menerbitkan beberapa buku. Olnes juga aktif mendapatkan hibah penelitian dan pengabdian masyarakat dari berbagai sumber. Selain itu, ia mendukung program pemerintah sebagai Dosen Pembimbing Lapangan (DPL) untuk program Kampus Mengajar dan kegiatan sertifikasi kompetensi. Ia juga berperan sebagai asesor BNSP dan Tim Penilai Ahli di Kementerian PUPR untuk bidang MEP. Untuk informasi lebih lanjut, Olnes dapat dihubungi melalui email olnes.hutajulu@unimed.ac.id atau nestajulu@gmail.com.

Penulis 5



Muhammad Dominique Mendoza adalah seorang akademisi dan peneliti di bidang teknologi pendidikan yang saat ini berprofesi sebagai dosen di Universitas Negeri Medan. Dengan latar belakang pendidikan yang kuat, ia meraih gelar S1 Double Degree dalam Sistem Informasi dan Manajemen dari Universitas Bina Nusantara, serta menyelesaikan studi S2 Manajemen di Universitas

Sumatera Utara. Minat penelitiannya berfokus pada Mixed Reality dalam pendidikan dan manajemen teknologi, yang telah membawanya meraih hibah penelitian setiap tahun dari universitas maupun kementerian. Berkat dedikasinya, berbagai hasil penelitiannya telah dipublikasikan di jurnal terakreditasi nasional (SINTA) dan jurnal internasional bereputasi (Scopus). Untuk kolaborasi atau informasi lebih lanjut, ia dapat dihubungi melalui email aenaen@unimed.ac.id, atau diikuti di Instagram: [@dmq_mendoza](https://www.instagram.com/dmq_mendoza).



**UNIVERSITAS
PRIMA INDONESIA**
UNPRI PRESS